**Experiment No. 5**

NAME:Omkar Khanvilkar ROLL NO: 07

CLASS: TY\_IT-B BATCH: 2

DATE OF PERFORMANCE: 20/08/2024 PRN No: 12210313

**Problem Statement:**

**Implement multithreading for Matrix Operations using Pthreads.**

We will implement the following Matrix Operations:

**1) Matrix Addition :**

The dimensions of both the matrices(operands) must be same to perform addition.

**2) Matrix Subtraction**

The dimensions of both the matrices(operands) must be same to perform addition.

**3) Matrix Multiplication**

The number of columns in the first matrix must be equal to the number of rows in the second matrix to perform multiplication.

# Matrix Addition

## Code:

#include <stdio.h>

#include <pthread.h>

#define N 3

int A[N][N], B[N][N], C[N][N];

void\* add\_matrices(void\* arg) {

    int thread\_id = \*(int\*)arg;

    for (int i = thread\_id; i < N; i += 2) {

        for (int j = 0; j < N; j++) {

            C[i][j] = A[i][j] + B[i][j];

        }

    }

    return NULL;

}

void printMatrix(int matrix[N][N]){

    for(int i = 0; i < N; i++){

        for(int j = 0; j < N; j++){

            printf("%d ",matrix[i][j]);

        }

        printf("\n");

    }

}

int main() {

*// Initialize matrices A , B and C*

    for (int i = 0; i < N; i++) {

        for (int j = 0; j < N; j++) {

            A[i][j] = i + j;

            B[i][j] = i - j;

            C[i][j] = 0;

        }

    }

*// print matrices A and B*

    printf("Matrix A:\n");

    printMatrix(A);

    printf("Matrix B:\n");

    printMatrix(B);

    pthread\_t threads[2];

    int thread\_ids[2] = {0, 1};

    for (int i = 0; i < 2; i++) {

        pthread\_create(&threads[i], NULL, add\_matrices, &thread\_ids[i]);

    }

    for (int i = 0; i < 2; i++) {

        pthread\_join(threads[i], NULL);

    }

    printf("Addition of Matrix A and B is Matrix C: \n");

    printMatrix(C);

    return 0;

}

## Output:

# **A computer screen with white text Description automatically generated**

# Matrix Subtraction

## Code:

#include <stdio.h>

#include <pthread.h>

#define N 3

int A[N][N], B[N][N], C[N][N];

void\* subtract\_matrices(void\* arg) {

    int thread\_id = \*(int\*)arg;

    for (int i = thread\_id; i < N; i += 2) {

        for (int j = 0; j < N; j++) {

            C[i][j] = A[i][j] - B[i][j];

        }

    }

    return NULL;

}

void printMatrix(int matrix[N][N]){

    for(int i = 0; i < N; i++){

        for(int j = 0; j < N; j++){

            printf("%d ",matrix[i][j]);

        }

        printf("\n");

    }

}

int main() {

*// Initialize matrices A , B and C*

    for (int i = 0; i < N; i++) {

        for (int j = 0; j < N; j++) {

            A[i][j] = i + j;

            B[i][j] = i - j;

            C[i][j] = 0;

        }

    }

*// print matrices A and B*

    printf("Matrix A:\n");

    printMatrix(A);

    printf("Matrix B:\n");

    printMatrix(B);

    pthread\_t threads[2];

    int thread\_ids[2] = {0, 1};

    for (int i = 0; i < 2; i++) {

        pthread\_create(&threads[i], NULL, subtract\_matrices, &thread\_ids[i]);

    }

    for (int i = 0; i < 2; i++) {

        pthread\_join(threads[i], NULL);

    }

    printf("Subtraction of Matrix A and B is Matrix C: \n");

    printMatrix(C);

    return 0;

}

## Output:

**A computer screen with white text

Description automatically generated**

# Matrix Multiplication

## Code:

#include <stdio.h>

#include <pthread.h>

#define N 3

int A[N][N], B[N][N], C[N][N];

void\* multiply\_matrices(void\* arg) {

    int thread\_id = \*(int\*)arg;

    for (int i = thread\_id; i < N; i += 2) {

        for (int j = 0; j < N; j++) {

            for(int k = 0; k < N; k++){

                C[i][j] += A[i][k] \* B[k][j];

            }

        }

    }

    return NULL;

}

void printMatrix(int matrix[N][N]){

    for(int i = 0; i < N; i++){

        for(int j = 0; j < N; j++){

            printf("%d ",matrix[i][j]);

        }

        printf("\n");

    }

}

int main() {

*// Initialize matrices A , B and C*

    for (int i = 0; i < N; i++) {

        for (int j = 0; j < N; j++) {

            A[i][j] = i + j;

            B[i][j] = i - j;

            C[i][j] = 0;

        }

    }

*// print matrices A and B*

    printf("Matrix A:\n");

    printMatrix(A);

    printf("Matrix B:\n");

    printMatrix(B);

    pthread\_t threads[2];

    int thread\_ids[2] = {0, 1};

    for (int i = 0; i < 2; i++) {

        pthread\_create(&threads[i], NULL, multiply\_matrices, &thread\_ids[i]);

    }

    for (int i = 0; i < 2; i++) {

        pthread\_join(threads[i], NULL);

    }

    printf("Multiplication of Matrix A and B is Matrix C: \n");

    printMatrix(C);

    return 0;

}

## Output:

