Assignment 3

Matrix multiplication with multithreading using pthreads

Code:

```c
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <stdlib.h>
#define MAX 50

void *multiply(void *arg);

int main()
{

   int mat1[MAX][MAX];
   int mat2[MAX][MAX];

   int row1, col1, row2, col2, i, j, k;
   printf("Enter the size of of matrix 1: ");
   scanf("%d %d", &row1, &col1);

   printf("Enter matrix 1 : \n");
   for (i = 0; i < row1; i++)
       for (j = 0; j < col1; j++)
           scanf("%d", &mat1[i][j]);

   printf("Enter the size of matrix 2: ");
   scanf("%d %d", &row2, &col2);

   printf("Enter matrix 2 : \n");
   for (i = 0; i < row1; i++)
       for (j = 0; j < col1; j++)
           scanf("%d", &mat2[i][j]);

   int max = row1 * col2;

   pthread_t threads[max];
```

```
    int nThread = 0;
    int *data = NULL;
    for (i = 0; i < row1; i++)
    {
        for (j = 0; j < col2; j++)
        {

            data = (int *)malloc((20) * sizeof(int));
            data[0] = col1;

            for (k = 0; k < col1; k++)
                data[k + 1] = mat1[i][k];

            for (k = 0; k < row2; k++)
                data[k + col1 + 1] = mat2[k][j];

            pthread_create(&threads[nThread++], NULL, multiply, (void
*)(data));
        }
    }
    printf("Thread clount: %d\n\n", nThread);
    printf("Multiplied Matrix : \n");
    for (i = 0; i < max; i++)
    {
        void *k;

        pthread_join(threads[i], &k);

        int *p = (int *)k;
        printf("%d ", *p);
        if ((i + 1) % col2 == 0)
            printf("\n");
    }

    return 0;
}

void *multiply(void *arg)
{
```

```c
    int *data = (int *)arg;
    int k = 0, i = 0;

    int x = data[0];
    for (i = 1; i <= x; i++)
        k += data[i] * data[i + x];

    int *p = (int *)malloc(sizeof(int));
    *p = k;

    pthread_exit(p);
}
```

Output:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL
→  a3 gcc code.c -lpthread && ./a.out
Enter the size of of matrix 1: 2 2
Enter matrix 1 :
1 2
3 4
Enter the size of matrix 2: 2 3
Enter matrix 2 :
2 4 1
8 4 0
Thread clount: 6

Multiplied Matrix :
4 20 0
10 44 0
→  a3
```