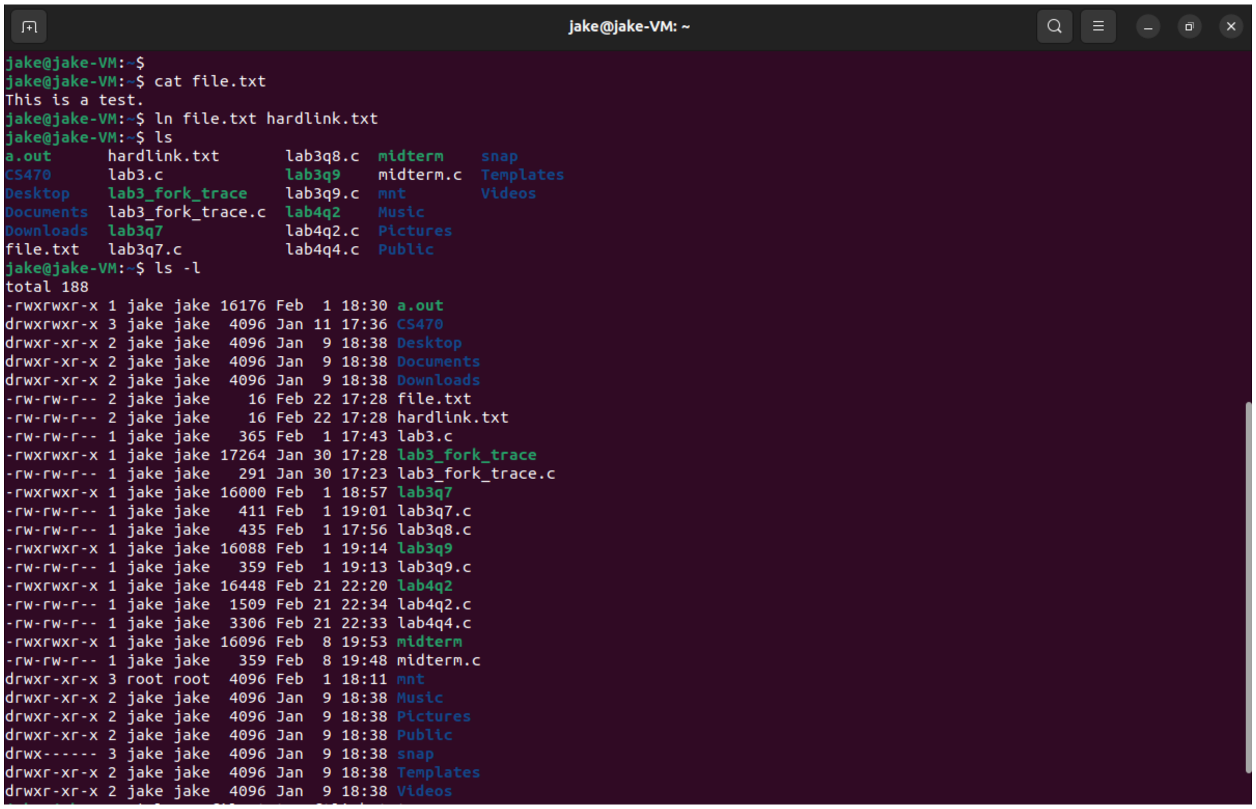
Robinson – CS 470 Lab 4: C Threads

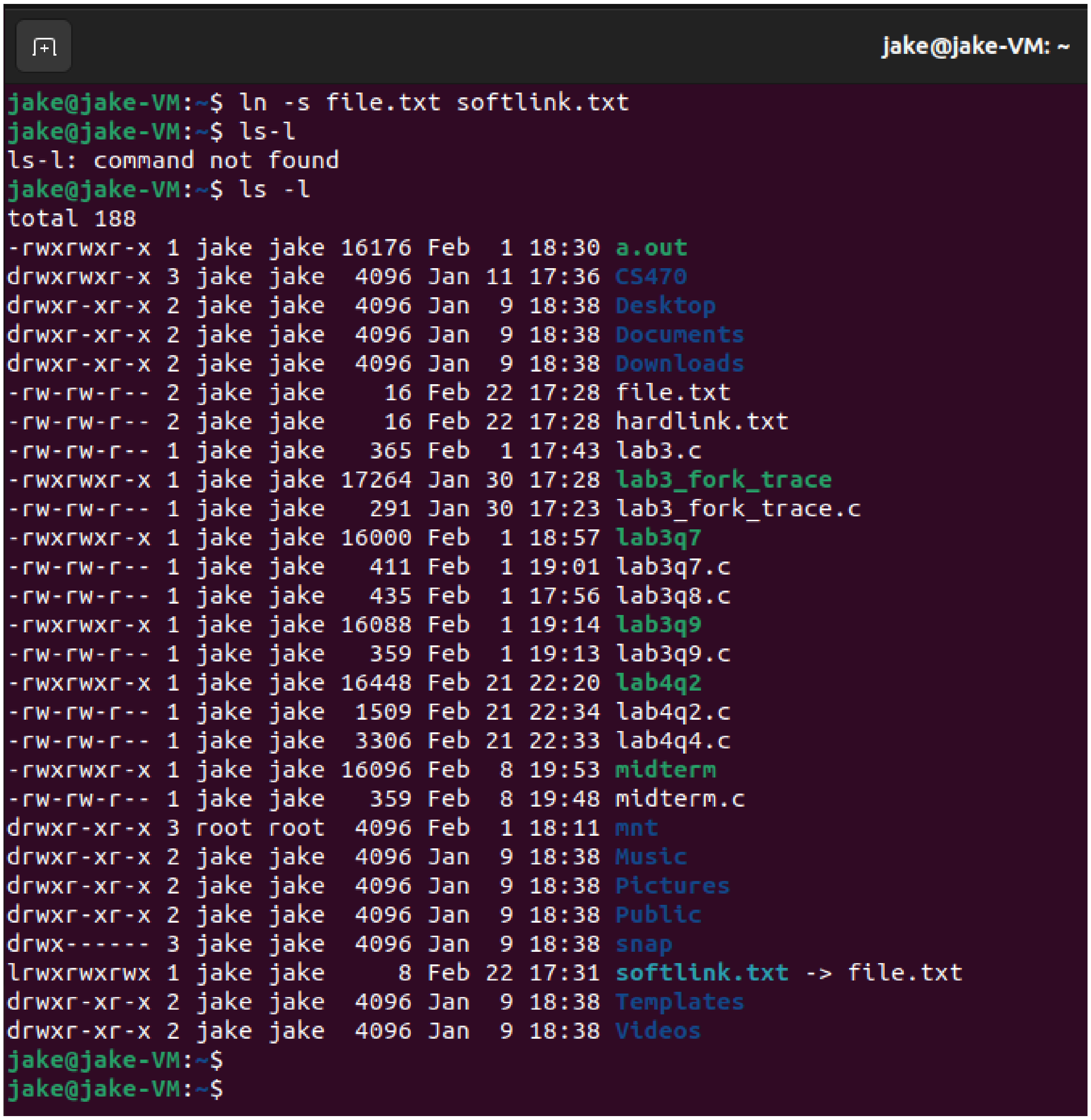
1. Code:



jake@jake-VM: ~

```
jake@jake-VM:~$
jake@jake-VM:~$ cat file.txt
This is a test.
jake@jake-VM:~$ ln file.txt hardlink.txt
jake@jake-VM:~$ ls
a.out      hardlink.txt       lab3q8.c  midterm    snap
CS470      lab3.c             lab3q9    midterm.c  Templates
Desktop    lab3_fork_trace    lab3q9.c  mnt        Videos
Documents  lab3_fork_trace.c  lab4q2    Music
Downloads  lab3q7             lab4q2.c  Pictures
file.txt   lab3q7.c           lab4q4.c  Public
jake@jake-VM:~$ ls -l
total 188
-rwxrwxr-x 1 jake jake 16176 Feb  1 18:30 a.out
drwxrwxr-x 3 jake jake  4096 Jan 11 17:36 CS470
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Desktop
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Documents
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Downloads
-rw-rw-r-- 2 jake jake    16 Feb 22 17:28 file.txt
-rw-rw-r-- 2 jake jake    16 Feb 22 17:28 hardlink.txt
-rw-rw-r-- 1 jake jake   365 Feb  1 17:43 lab3.c
-rwxrwxr-x 1 jake jake 17264 Jan 30 17:28 lab3_fork_trace
-rw-rw-r-- 1 jake jake   291 Jan 30 17:23 lab3_fork_trace.c
-rwxrwxr-x 1 jake jake 16000 Feb  1 18:57 lab3q7
-rw-rw-r-- 1 jake jake   411 Feb  1 19:01 lab3q7.c
-rw-rw-r-- 1 jake jake   435 Feb  1 17:56 lab3q8.c
-rwxrwxr-x 1 jake jake 16088 Feb  1 19:14 lab3q9
-rw-rw-r-- 1 jake jake   359 Feb  1 19:13 lab3q9.c
-rwxrwxr-x 1 jake jake 16448 Feb 21 22:20 lab4q2
-rw-rw-r-- 1 jake jake  1509 Feb 21 22:34 lab4q2.c
-rw-rw-r-- 1 jake jake  3306 Feb 21 22:33 lab4q4.c
-rwxrwxr-x 1 jake jake 16096 Feb  8 19:53 midterm
-rw-rw-r-- 1 jake jake   359 Feb  8 19:48 midterm.c
drwxr-xr-x 3 root root  4096 Feb  1 18:11 mnt
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Music
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Pictures
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Public
drwx------ 3 jake jake  4096 Jan  9 18:38 snap
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Templates
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Videos
```

Softlink:

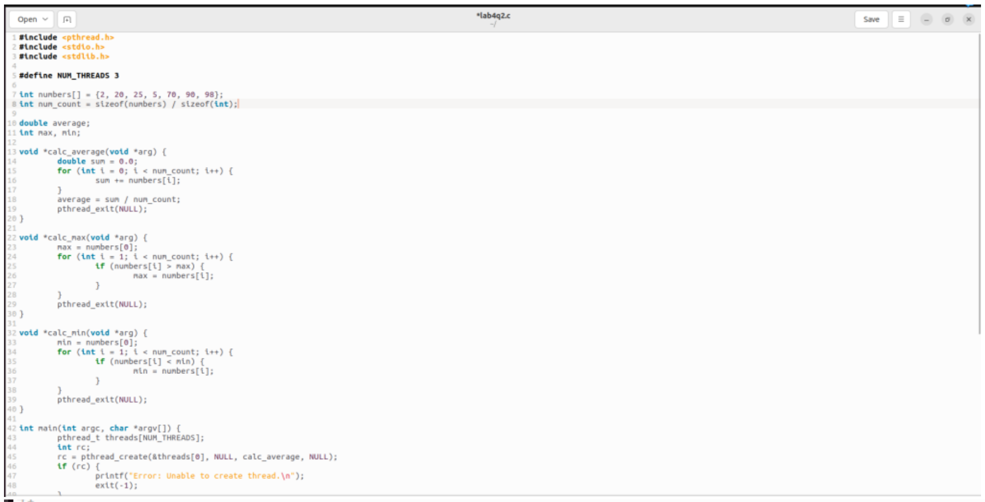


```
jake@jake-VM:~$ ln -s file.txt softlink.txt
jake@jake-VM:~$ ls-l
ls-l: command not found
jake@jake-VM:~$ ls -l
total 188
-rwxrwxr-x 1 jake jake 16176 Feb  1 18:30 a.out
drwxrwxr-x 3 jake jake  4096 Jan 11 17:36 CS470
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Desktop
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Documents
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Downloads
-rw-rw-r-- 2 jake jake    16 Feb 22 17:28 file.txt
-rw-rw-r-- 2 jake jake    16 Feb 22 17:28 hardlink.txt
-rw-rw-r-- 1 jake jake   365 Feb  1 17:43 lab3.c
-rwxrwxr-x 1 jake jake 17264 Jan 30 17:28 lab3_fork_trace
-rw-rw-r-- 1 jake jake   291 Jan 30 17:23 lab3_fork_trace.c
-rwxrwxr-x 1 jake jake 16000 Feb  1 18:57 lab3q7
-rw-rw-r-- 1 jake jake   411 Feb  1 19:01 lab3q7.c
-rw-rw-r-- 1 jake jake   435 Feb  1 17:56 lab3q8.c
-rwxrwxr-x 1 jake jake 16088 Feb  1 19:14 lab3q9
-rw-rw-r-- 1 jake jake   359 Feb  1 19:13 lab3q9.c
-rwxrwxr-x 1 jake jake 16448 Feb 21 22:20 lab4q2
-rw-rw-r-- 1 jake jake  1509 Feb 21 22:34 lab4q2.c
-rw-rw-r-- 1 jake jake  3306 Feb 21 22:33 lab4q4.c
-rwxrwxr-x 1 jake jake 16096 Feb  8 19:53 midterm
-rw-rw-r-- 1 jake jake   359 Feb  8 19:48 midterm.c
drwxr-xr-x 3 root root  4096 Feb  1 18:11 mnt
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Music
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Pictures
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Public
drwx------ 3 jake jake  4096 Jan  9 18:38 snap
lrwxrwxrwx 1 jake jake     8 Feb 22 17:31 softlink.txt -> file.txt
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Templates
drwxr-xr-x 2 jake jake  4096 Jan  9 18:38 Videos
jake@jake-VM:~$
jake@jake-VM:~$
```

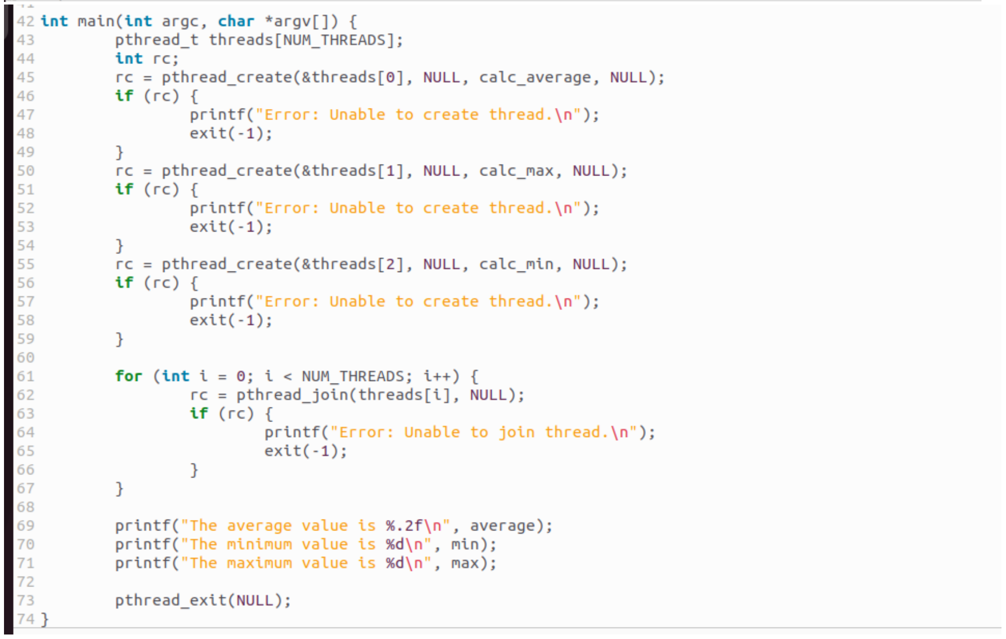
Summary of Code:

Using ln on an existing text file we can make a new text file creating a hardlink to both files. We can repeat this process again but with ln -s to create a softlink text file.

2. Code:

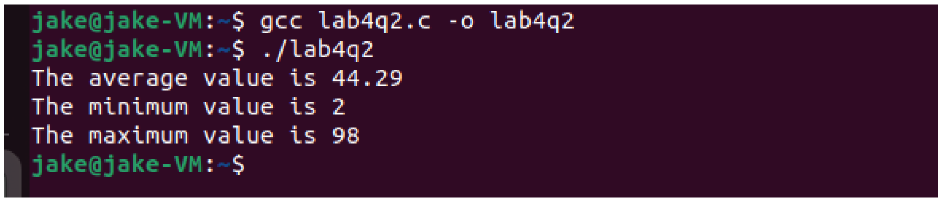


```c
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>

#define NUM_THREADS 3

int numbers[] = {2, 20, 25, 5, 70, 90, 98};
int num_count = sizeof(numbers) / sizeof(int);

double average;
int max, min;

void *calc_average(void *arg) {
        double sum = 0.0;
        for (int i = 0; i < num_count; i++) {
                sum += numbers[i];
        }
        average = sum / num_count;
        pthread_exit(NULL);
}

void *calc_max(void *arg) {
        max = numbers[0];
        for (int i = 1; i < num_count; i++) {
                if (numbers[i] > max) {
                        max = numbers[i];
                }
        }
        pthread_exit(NULL);
}

void *calc_min(void *arg) {
        min = numbers[0];
        for (int i = 1; i < num_count; i++) {
                if (numbers[i] < min) {
                        min = numbers[i];
                }
        }
        pthread_exit(NULL);
}

int main(int argc, char *argv[]) {
        pthread_t threads[NUM_THREADS];
        int rc;
        rc = pthread_create(&threads[0], NULL, calc_average, NULL);
        if (rc) {
                printf("Error: Unable to create thread.\n");
                exit(-1);
```



```c
int main(int argc, char *argv[]) {
        pthread_t threads[NUM_THREADS];
        int rc;
        rc = pthread_create(&threads[0], NULL, calc_average, NULL);
        if (rc) {
                printf("Error: Unable to create thread.\n");
                exit(-1);
        }
        rc = pthread_create(&threads[1], NULL, calc_max, NULL);
        if (rc) {
                printf("Error: Unable to create thread.\n");
                exit(-1);
        }
        rc = pthread_create(&threads[2], NULL, calc_min, NULL);
        if (rc) {
                printf("Error: Unable to create thread.\n");
                exit(-1);
        }

        for (int i = 0; i < NUM_THREADS; i++) {
                rc = pthread_join(threads[i], NULL);
                if (rc) {
                        printf("Error: Unable to join thread.\n");
                        exit(-1);
                }
        }

        printf("The average value is %.2f\n", average);
        printf("The minimum value is %d\n", min);
        printf("The maximum value is %d\n", max);

        pthread_exit(NULL);
}
```

Output:

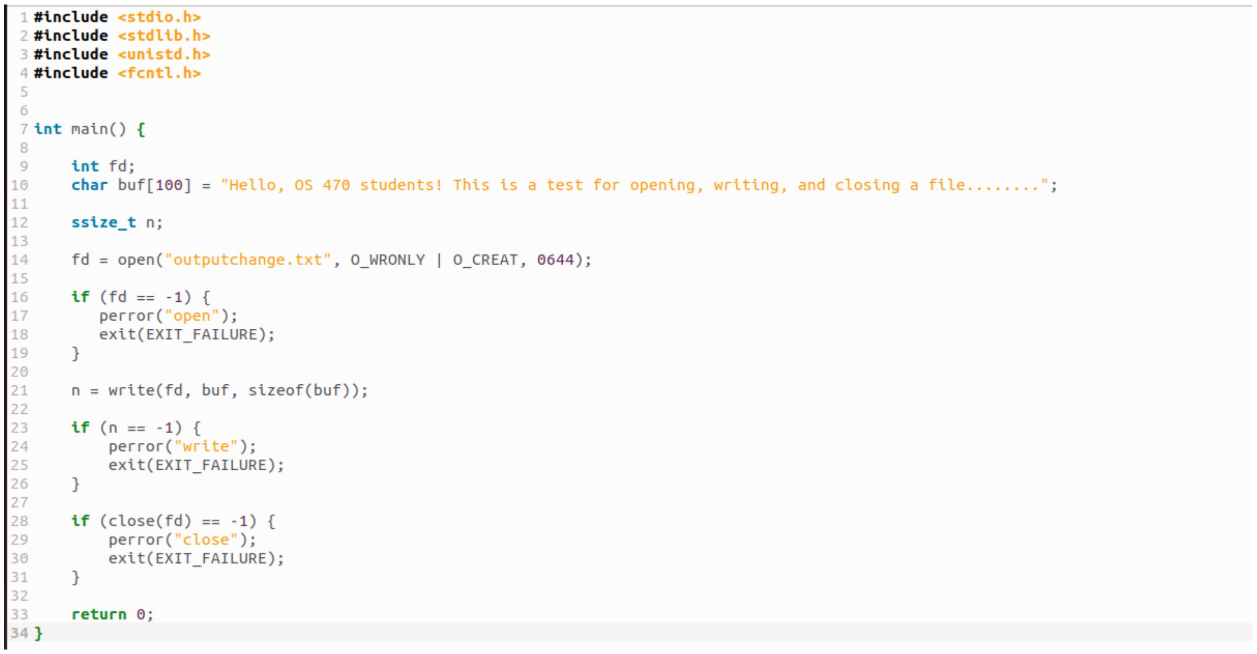


```
jake@jake-VM:~$ gcc lab4q2.c -o lab4q2
jake@jake-VM:~$ ./lab4q2
The average value is 44.29
The minimum value is 2
The maximum value is 98
jake@jake-VM:~$
```

Summary of Code:

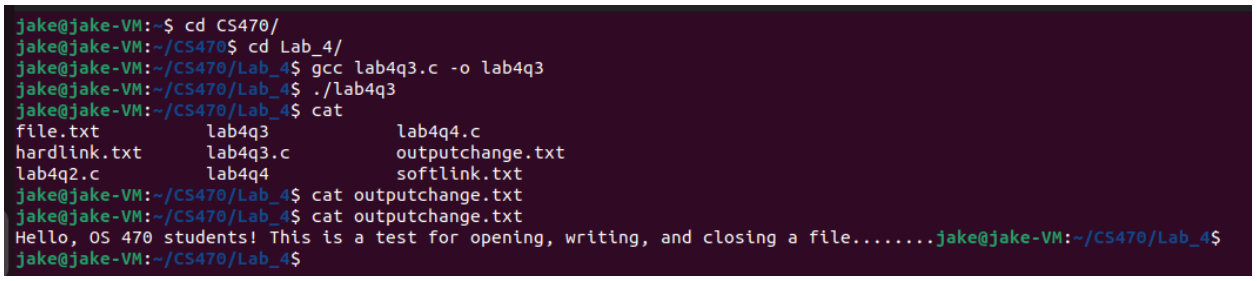
The code declares an array of pre-defined numbers along with several threads set to 3. We then have 3 functions for calculating the min, max, and average based off the defined number array and threads. We then create threads for min, max, and average and then join them together in a for loop.

3. Code:



```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>


int main() {

    int fd;
    char buf[100] = "Hello, OS 470 students! This is a test for opening, writing, and closing a file........";

    ssize_t n;

    fd = open("outputchange.txt", O_WRONLY | O_CREAT, 0644);

    if (fd == -1) {
       perror("open");
       exit(EXIT_FAILURE);
    }

    n = write(fd, buf, sizeof(buf));

    if (n == -1) {
        perror("write");
        exit(EXIT_FAILURE);
    }

    if (close(fd) == -1) {
        perror("close");
        exit(EXIT_FAILURE);
    }

    return 0;
}
```

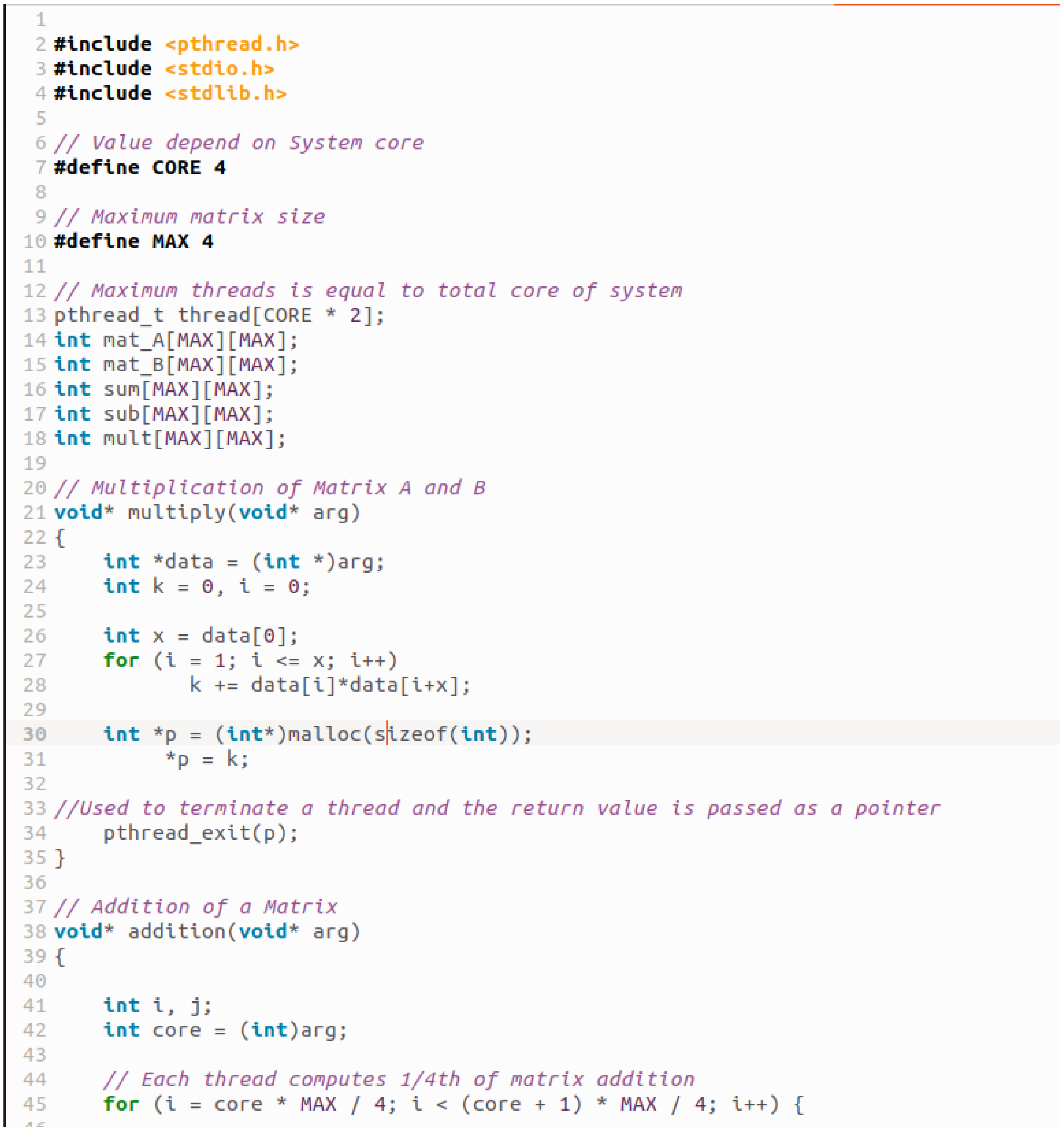
Output:



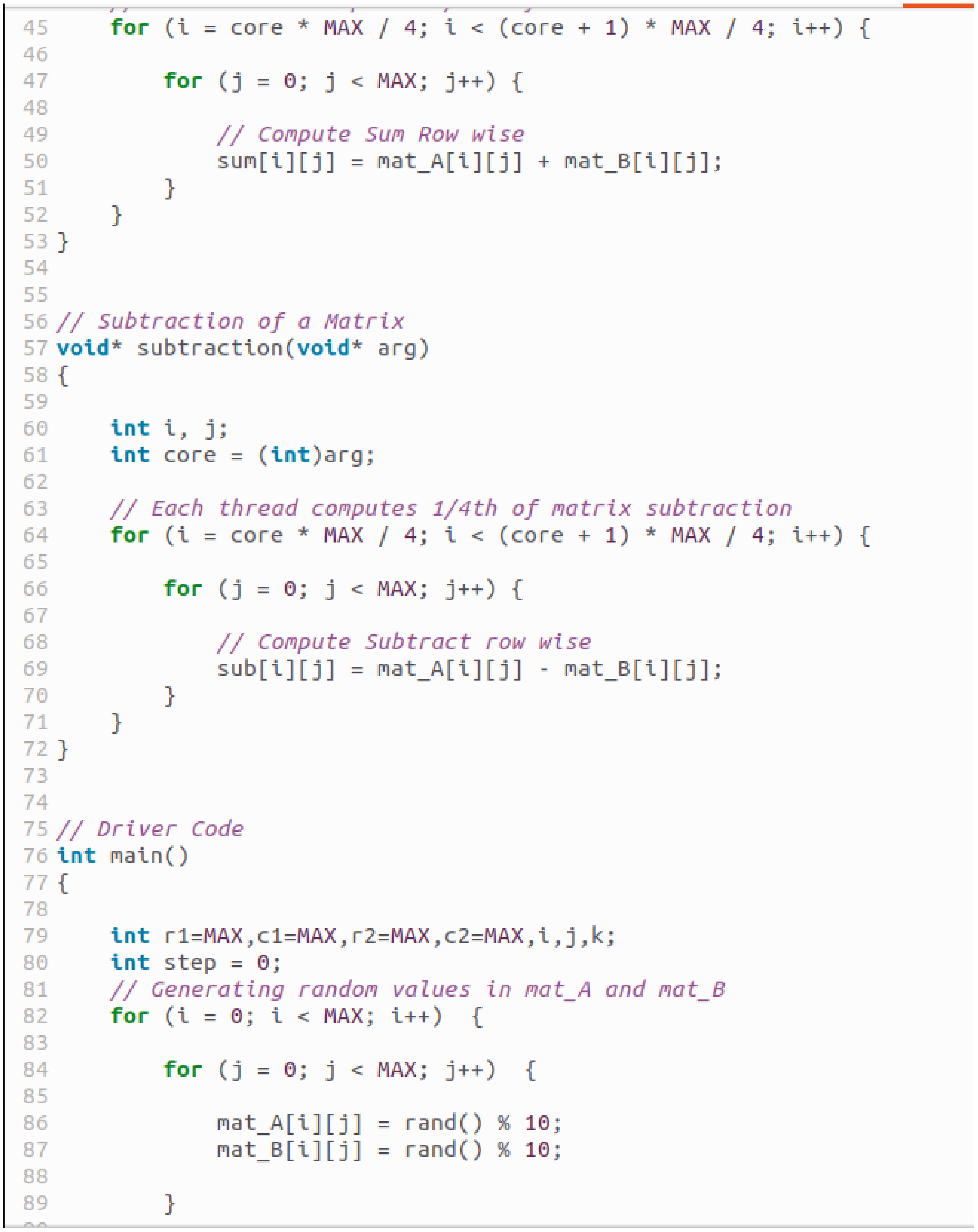
```
jake@jake-VM:~$ cd CS470/
jake@jake-VM:~/CS470$ cd Lab_4/
jake@jake-VM:~/CS470/Lab_4$ gcc lab4q3.c -o lab4q3
jake@jake-VM:~/CS470/Lab_4$ ./lab4q3
jake@jake-VM:~/CS470/Lab_4$ cat
file.txt          lab4q3            lab4q4.c
hardlink.txt      lab4q3.c          outputchange.txt
lab4q2.c          lab4q4            softlink.txt
jake@jake-VM:~/CS470/Lab_4$ cat outputchange.txt
jake@jake-VM:~/CS470/Lab_4$ cat outputchange.txt
Hello, OS 470 students! This is a test for opening, writing, and closing a file........jake@jake-VM:~/CS470/Lab_4$
jake@jake-VM:~/CS470/Lab_4$
```

Summary of Code:

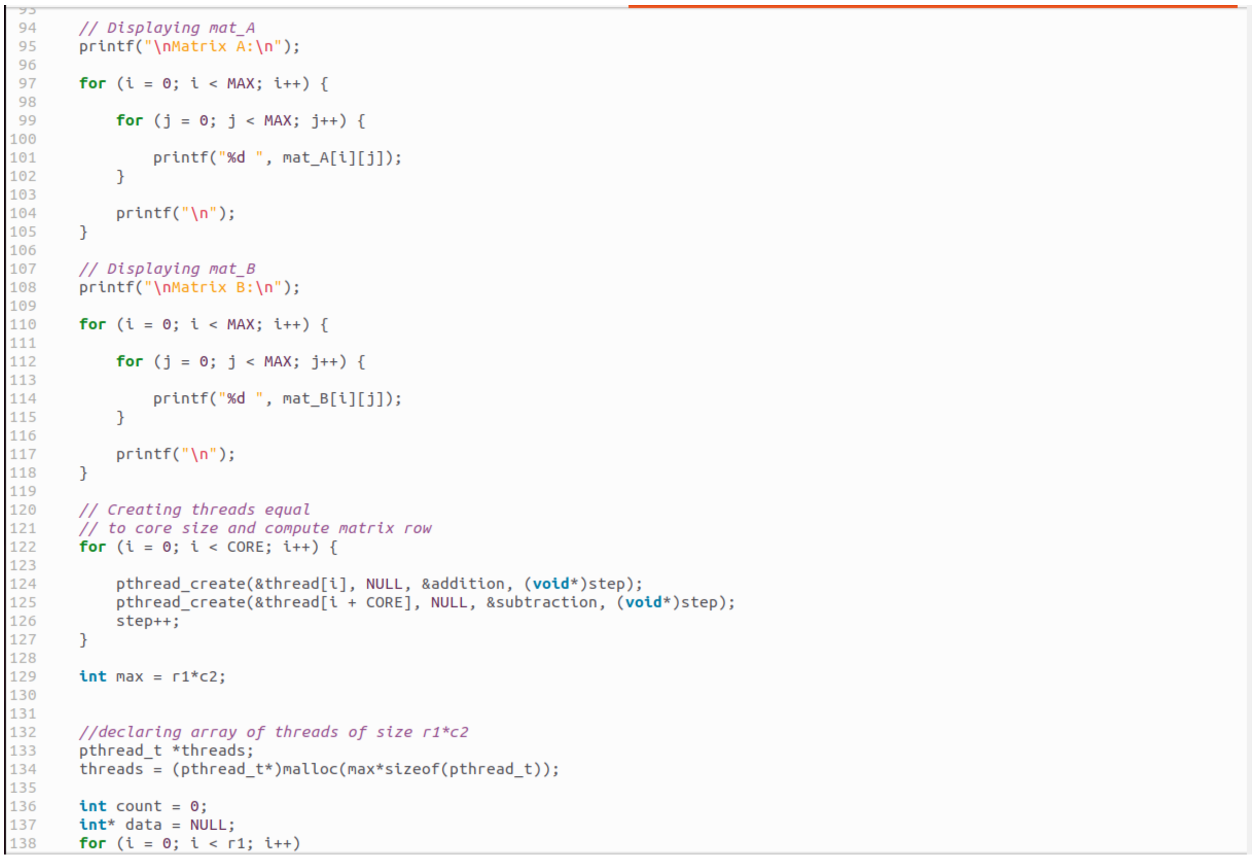
The code above uses a character buf and an int assignment to execute the commands to open write and close a file in a specific way. we open the file "outputchange.txt" using the open() system call with the O_WRONLY and O_CREAT flags. The O_WRONLY flag specifies that we are opening the file for writing only. The O_CREAT flag specifies that the file should be created if it does not already exist. The third argument, 0644, specifies the file permissions.

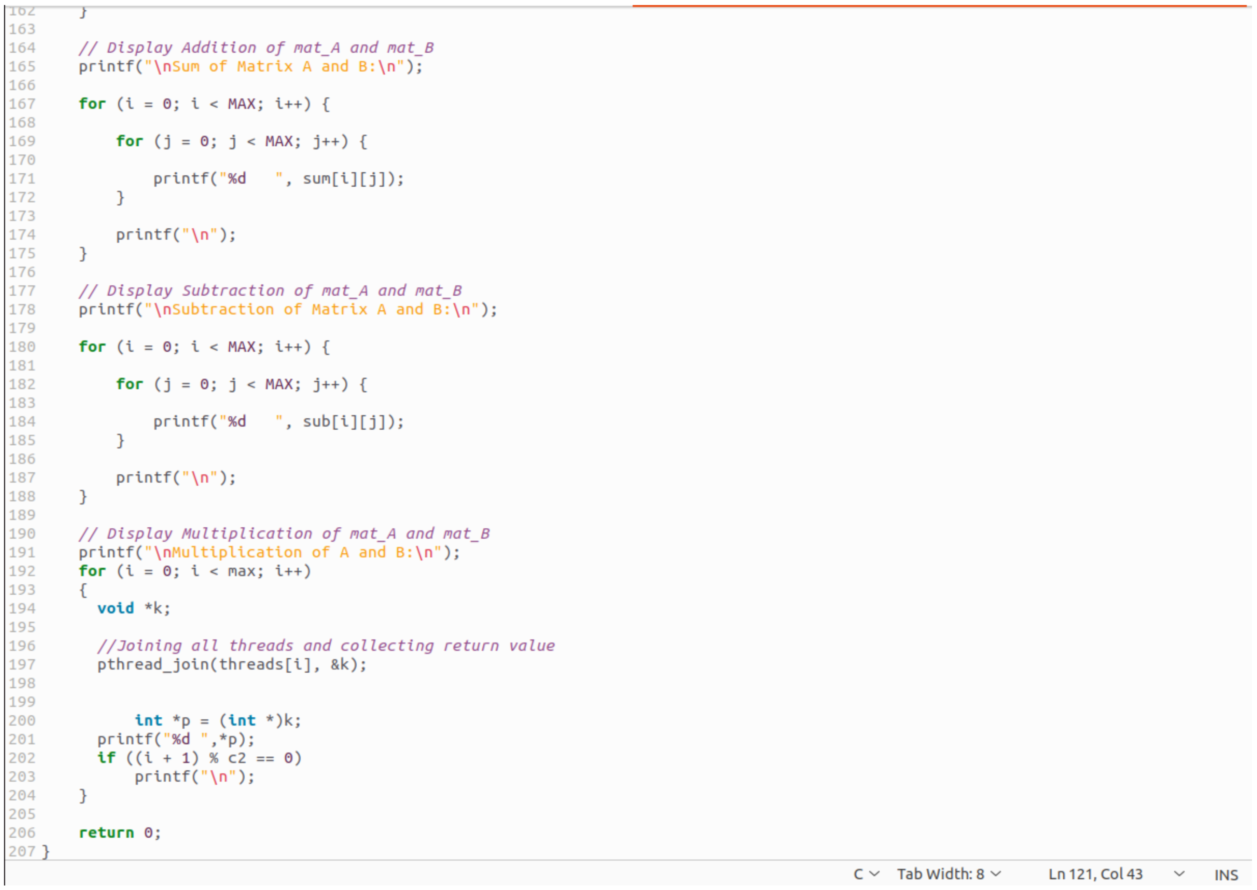


```c

#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>

// Value depend on System core
#define CORE 4

// Maximum matrix size
#define MAX 4

// Maximum threads is equal to total core of system
pthread_t thread[CORE * 2];
int mat_A[MAX][MAX];
int mat_B[MAX][MAX];
int sum[MAX][MAX];
int sub[MAX][MAX];
int mult[MAX][MAX];

// Multiplication of Matrix A and B
void* multiply(void* arg)
{
    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;

//Used to terminate a thread and the return value is passed as a pointer
    pthread_exit(p);
}

// Addition of a Matrix
void* addition(void* arg)
{

    int i, j;
    int core = (int)arg;

    // Each thread computes 1/4th of matrix addition
    for (i = core * MAX / 4; i < (core + 1) * MAX / 4; i++) {
```



```c
    for (i = core * MAX / 4; i < (core + 1) * MAX / 4; i++) {

        for (j = 0; j < MAX; j++) {

            // Compute Sum Row wise
            sum[i][j] = mat_A[i][j] + mat_B[i][j];
        }
    }
}


// Subtraction of a Matrix
void* subtraction(void* arg)
{

    int i, j;
    int core = (int)arg;

    // Each thread computes 1/4th of matrix subtraction
    for (i = core * MAX / 4; i < (core + 1) * MAX / 4; i++) {

        for (j = 0; j < MAX; j++) {

            // Compute Subtract row wise
            sub[i][j] = mat_A[i][j] - mat_B[i][j];
        }
    }
}


// Driver Code
int main()
{

    int r1=MAX,c1=MAX,r2=MAX,c2=MAX,i,j,k;
    int step = 0;
    // Generating random values in mat_A and mat_B
    for (i = 0; i < MAX; i++)  {

        for (j = 0; j < MAX; j++)  {

            mat_A[i][j] = rand() % 10;
            mat_B[i][j] = rand() % 10;

        }
```

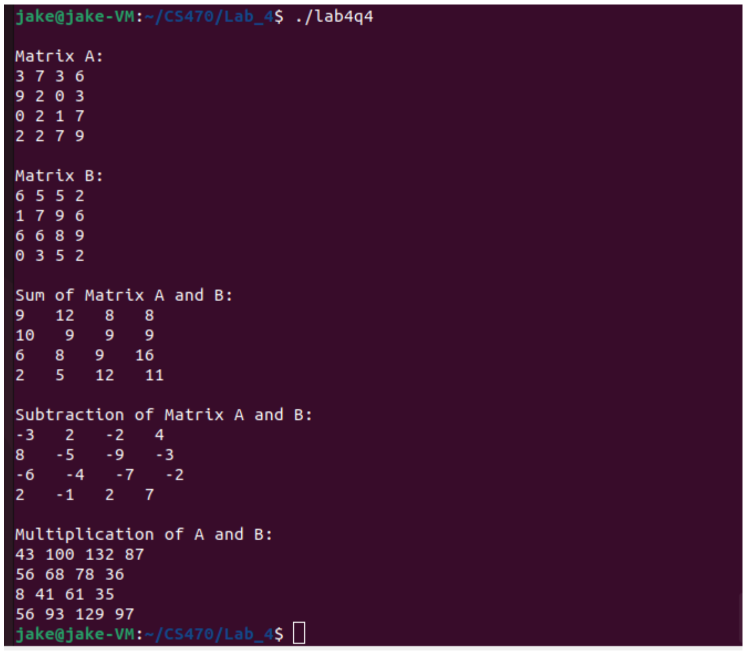

```c
    // Displaying mat_A
    printf("\nMatrix A:\n");

    for (i = 0; i < MAX; i++) {

        for (j = 0; j < MAX; j++) {

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

        printf("\n");
    }

    // Displaying mat_B
    printf("\nMatrix B:\n");

    for (i = 0; i < MAX; i++) {

        for (j = 0; j < MAX; j++) {

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

        printf("\n");
    }

    // Creating threads equal
    // to core size and compute matrix row
    for (i = 0; i < CORE; i++) {

        pthread_create(&thread[i], NULL, &addition, (void*)step);
        pthread_create(&thread[i + CORE], NULL, &subtraction, (void*)step);
        step++;
    }

    int max = r1*c2;


    //declaring array of threads of size r1*c2
    pthread_t *threads;
    threads = (pthread_t*)malloc(max*sizeof(pthread_t));

    int count = 0;
    int* data = NULL;
    for (i = 0; i < r1; i++)
```



```c
    }

    // Display Addition of mat_A and mat_B
    printf("\nSum of Matrix A and B:\n");

    for (i = 0; i < MAX; i++) {

        for (j = 0; j < MAX; j++) {

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

        printf("\n");
    }

    // Display Subtraction of mat_A and mat_B
    printf("\nSubtraction of Matrix A and B:\n");

    for (i = 0; i < MAX; i++) {

        for (j = 0; j < MAX; j++) {

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

        printf("\n");
    }

    // Display Multiplication of mat_A and mat_B
    printf("\nMultiplication of A and B:\n");
    for (i = 0; i < max; i++)
    {
      void *k;

      //Joining all threads and collecting return value
      pthread_join(threads[i], &k);


          int *p = (int *)k;
      printf("%d ",*p);
      if ((i + 1) % c2 == 0)
          printf("\n");
    }

    return 0;
}
```

C ⌄   Tab Width: 8 ⌄   Ln 121, Col 43   ⌄   INS

Output:



```
jake@jake-VM:~/CS470/Lab_4$ ./lab4q4

Matrix A:
3 7 3 6
9 2 0 3
0 2 1 7
2 2 7 9

Matrix B:
6 5 5 2
1 7 9 6
6 6 8 9
0 3 5 2

Sum of Matrix A and B:
9   12   8   8
10   9   9   9
6   8   9   16
2   5   12   11

Subtraction of Matrix A and B:
-3   2   -2   4
8   -5   -9   -3
-6   -4   -7   -2
2   -1   2   7

Multiplication of A and B:
43 100 132 87
56 68 78 36
8 41 61 35
56 93 129 97
jake@jake-VM:~/CS470/Lab_4$ 
```

Summary of Code:

The code above starts by declaring a new thread based off of 4 defined system cores and a matrixes declared as 4 by 4s and initialize 5 matrixes 2 originals and ones to store the addition subtraction and multiplication. There are then 3 function to add, sub, and mult the values from the thread after they are stored in the two original matrixes. We then initialize them in the main and use nested for loops to print the matrices.