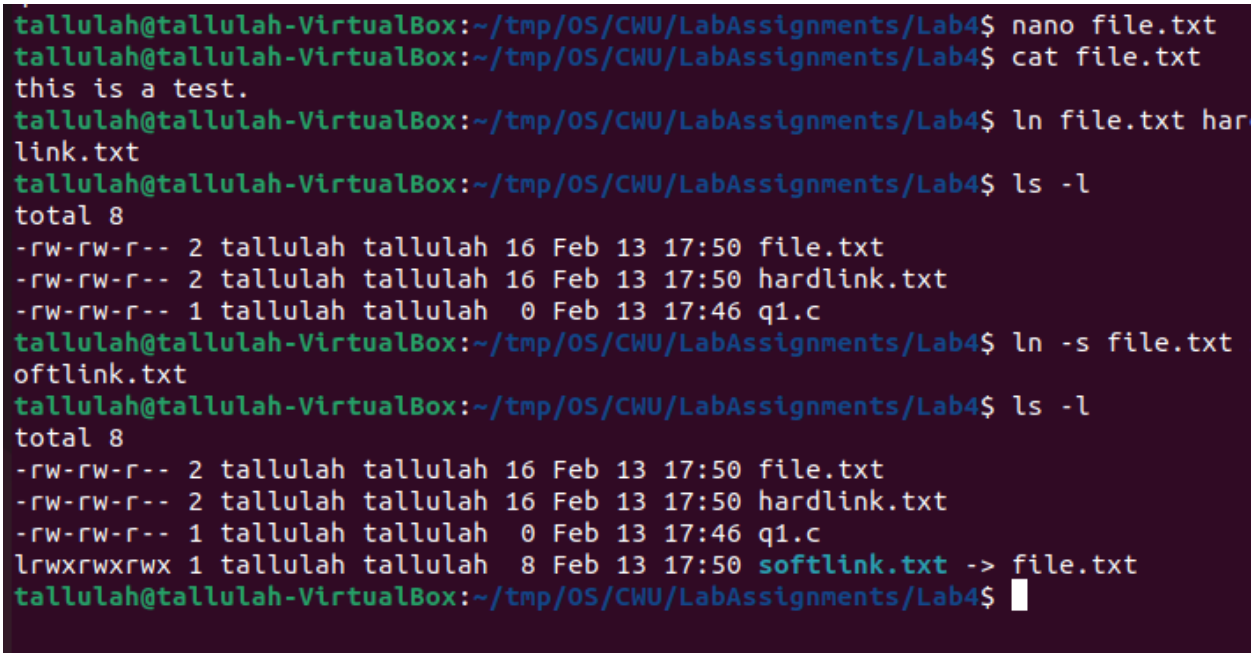
Question 1:



First I created a hardlink to file.txt, which created its own file as you can see in the first ls -l command. Then I created a softlink to file.txt, which basically created a pointer to the file, as you can see in the second ls -l command.

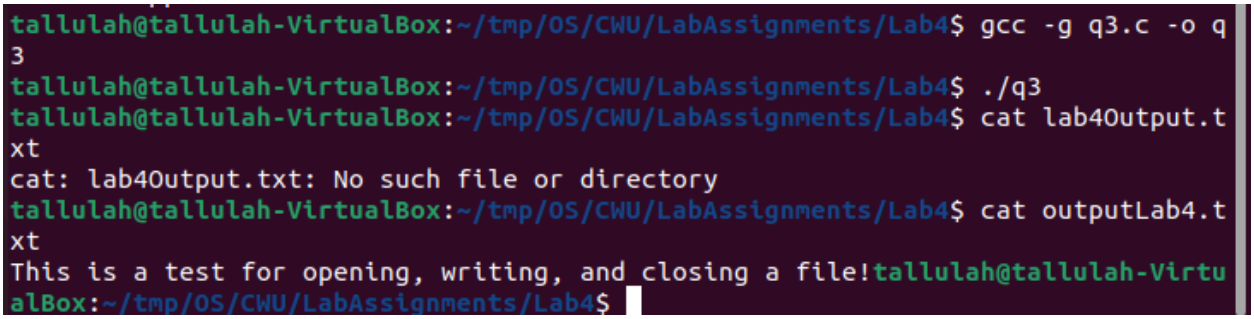
Question 2:

```
int main(int argc, char *argv[]){
        pthread_t threads[NUM_THREADS];
        int rc;
        rc = pthread_create(&threads[0], NULL, calc_max, NULL);
        if (rc){
                printf("Error, unable to create thread.\n");
                exit(-1);
        }
        rc = pthread_create(&threads[1], 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 maximum value is %d\n", max);
        printf("The minimum value is %d\n", min);

        pthread_exit(NULL);
```

```
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab4$ gcc q1.c -o q1
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab4$ ./q1
The maximum value is 95
The minimum value is 2
tallulah@tallulah-VirtualBox:~/tmp/OS/CWU/LabAssignments/Lab4$
```

For this one I passed in an array of integers and had two separate threads work through the array to determine the maximum and minimum values.

Question 3:



For this question I created a C file that opens a file called outputLab4 and writes to it the phrase “This is a test for opening, writing, and closing a file!”. To do this I used the open() system call to open the file. And I used the write() system call to write to the file. Then I used close() to close the file.

Question 4:

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>
#define CORE 4
#define MAX 4
pthread_t thread[CORE*3];
int mat_A[MAX][MAX], mat_B[MAX][MAX], sum[MAX][MAX], sub[MAX][MAX],
product[MAX][MAX];

void* addition(void* arg){
        int i,j;
        int core = (int)arg;

        for(i=core*MAX/4; i< (core+1)* MAX/4;i++){
                for(j=0; j< MAX; j++){
                        sum[i][j] = mat_A[i][j] + mat_B[i][j];
                }
        }
}
void* subtraction(void* arg){
        int i,j;
        int core = (int)arg;

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

```
Matrix A:
3736
9203
0217
2279

Matrix B:
6552
1796
6689
0352

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

```
Difference of Matrix A and B:
-3   2   -2   4
8   5   -9   -3
-6   -4   -7   -2
2   -1   2   7
Product of Matrix A and B:
18   35   15   12
9   14   0   18
0   12   8   63
0   6   35   18
```

To do this question I first created 2 matrices in the main method, populating the two with values under 10 for simplicity. Then I created threads equal to the core size and computed the matrix rows. The three different computations were done by having each thread compute ¼ of the matrix operation performed. After joining threads post computation, I displayed the three new matrices created for addition, subtraction, and multiplication.