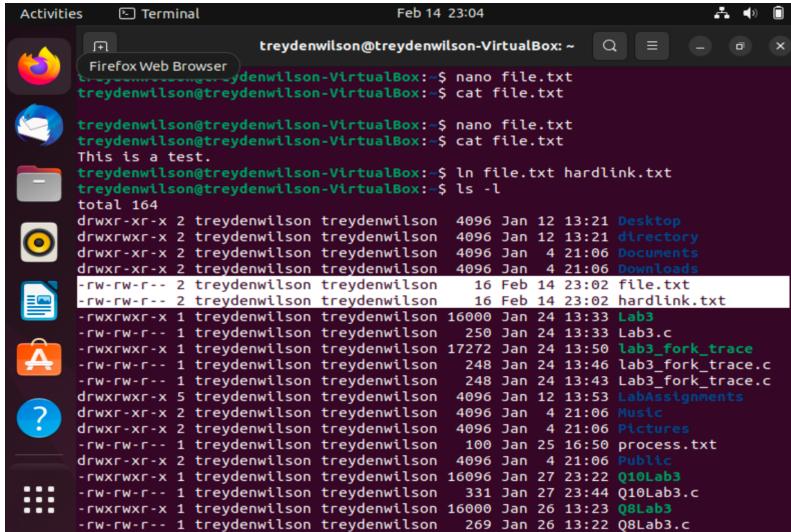


## 1. Using the ln and ln-s commands, create hard and soft links.

### Hard Link:

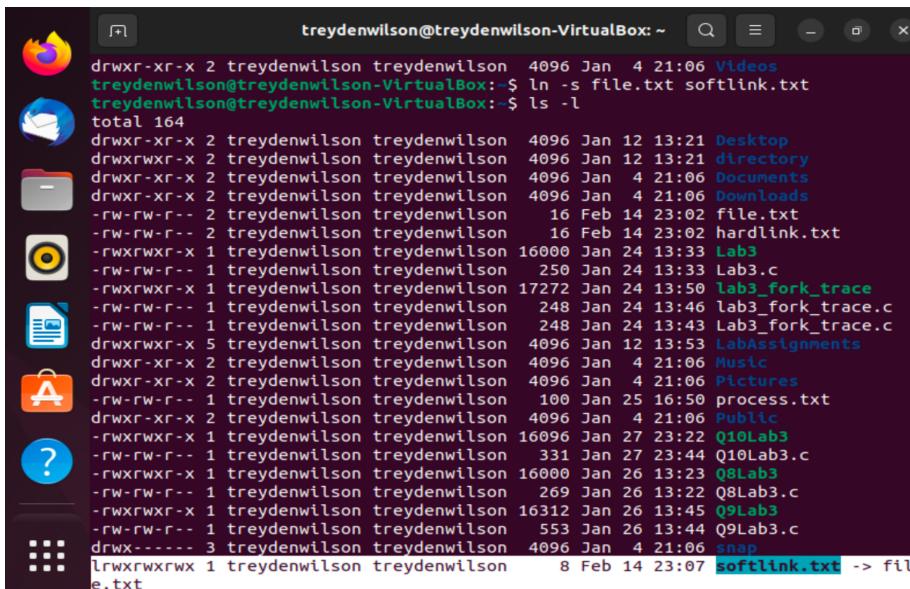


A screenshot of a Linux desktop environment showing a terminal window. The terminal window title is "treydenwilson@treydenwilson-VirtualBox: ~". The terminal content shows the following command sequence:

```
treydenwilson@treydenwilson-VirtualBox: ~$ nano file.txt
treydenwilson@treydenwilson-VirtualBox: ~$ cat file.txt
This is a test.
treydenwilson@treydenwilson-VirtualBox: ~$ ln file.txt hardlink.txt
treydenwilson@treydenwilson-VirtualBox: ~$ ls -l
total 164
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 12 13:21 Desktop
drwxrwxr-x 2 treydenwilson treydenwilson 4096 Jan 12 13:21 directory
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Documents
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Downloads
-rw-rw-r-- 2 treydenwilson treydenwilson 16 Feb 14 23:02 file.txt
-rw-rw-r-- 2 treydenwilson treydenwilson 16 Feb 14 23:02 hardlink.txt
-rwxrwxr-x 1 treydenwilson treydenwilson 16000 Jan 24 13:33 Lab3
-rw-rw-r-- 1 treydenwilson treydenwilson 250 Jan 24 13:33 Lab3.c
-rwxrwxr-x 1 treydenwilson treydenwilson 17272 Jan 24 13:50 lab3_fork_trace
-rw-rw-r-- 1 treydenwilson treydenwilson 248 Jan 24 13:46 lab3_fork_trace.c
-rw-rw-r-- 1 treydenwilson treydenwilson 248 Jan 24 13:43 lab3_fork_trace.c
drwxrwxr-x 5 treydenwilson treydenwilson 4096 Jan 12 13:53 LabAssignments
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Music
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Pictures
-rw-rw-r-- 1 treydenwilson treydenwilson 100 Jan 25 16:50 process.txt
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Public
-rwxrwxr-x 1 treydenwilson treydenwilson 16096 Jan 27 23:22 Q10Lab3
-rw-rw-r-- 1 treydenwilson treydenwilson 331 Jan 27 23:44 Q10Lab3.c
-rwxrwxr-x 1 treydenwilson treydenwilson 16000 Jan 26 13:23 Q8Lab3
-rw-rw-r-- 1 treydenwilson treydenwilson 269 Jan 26 13:22 Q8Lab3.c
```

The following code creates a hard link between file.txt and hardlink.txt. Furthermore, it shows that the original file and hard link have two hard links pointing to them. Finally, we verified that the hard link is indeed there, and the above screenshot proves that.

### Soft Link:



A screenshot of a Linux desktop environment showing a terminal window. The terminal window title is "treydenwilson@treydenwilson-VirtualBox: ~". The terminal content shows the following command sequence:

```
treydenwilson@treydenwilson-VirtualBox: ~$ ln -s file.txt softlink.txt
treydenwilson@treydenwilson-VirtualBox: ~$ ls -l
total 164
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 12 13:21 Desktop
drwxrwxr-x 2 treydenwilson treydenwilson 4096 Jan 12 13:21 directory
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Documents
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Downloads
-rw-rw-r-- 2 treydenwilson treydenwilson 16 Feb 14 23:02 file.txt
-rw-rw-r-- 2 treydenwilson treydenwilson 16 Feb 14 23:02 hardlink.txt
-rwxrwxr-x 1 treydenwilson treydenwilson 16000 Jan 24 13:33 Lab3
-rw-rw-r-- 1 treydenwilson treydenwilson 250 Jan 24 13:33 Lab3.c
-rwxrwxr-x 1 treydenwilson treydenwilson 17272 Jan 24 13:50 lab3_fork_trace
-rw-rw-r-- 1 treydenwilson treydenwilson 248 Jan 24 13:46 lab3_fork_trace.c
-rw-rw-r-- 1 treydenwilson treydenwilson 248 Jan 24 13:43 lab3_fork_trace.c
drwxrwxr-x 5 treydenwilson treydenwilson 4096 Jan 12 13:53 LabAssignments
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Music
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Pictures
-rw-rw-r-- 1 treydenwilson treydenwilson 100 Jan 25 16:50 process.txt
drwxr-xr-x 2 treydenwilson treydenwilson 4096 Jan 4 21:06 Public
-rwxrwxr-x 1 treydenwilson treydenwilson 16096 Jan 27 23:22 Q10Lab3
-rw-rw-r-- 1 treydenwilson treydenwilson 331 Jan 27 23:44 Q10Lab3.c
-rwxrwxr-x 1 treydenwilson treydenwilson 16000 Jan 26 13:23 Q8Lab3
-rw-rw-r-- 1 treydenwilson treydenwilson 269 Jan 26 13:22 Q8Lab3.c
drwx----- 3 treydenwilson treydenwilson 4096 Jan 4 21:06 snap
lrwxrwxrwx 1 treydenwilson treydenwilson 8 Feb 14 23:07 softlink.txt -> file.txt
```

The following code creates a soft link using the ln -s command with file.txt and softlink.txt. Then ls -l is used to view the soft link, which is shown at the bottom of the screenshot. You can see the soft link highlighted in white above.

- 2. Create a multithreaded program that computes different statistical values for a set of numbers.** When given a series of numbers on the command line, this application will start two independent worker threads. One thread will compute the greatest value, and the next will compute the minimum value. Assume your program is given a list of integers. (The array of numbers must be provided as a parameter to the threads, and the thread must return the calculated value to the main thread.)

```

1 #include <pthread.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4 #define NUM_THREADS 3
5
6 int numbers[] = {2, 20, 25, 5, 70, 90, 98};
7
8 int num_count = sizeof(numbers) / sizeof(int);
9 double average;
10 int max, min;
11
12 void *calc_average(void *arg) {
13     double sum = 0.0;
14     for(int i = 0; i < num_count; i++) {
15         sum += numbers[i];
16     }
17     average = sum / num_count;
18     pthread_exit(NULL);
19 }
20
21 }
22
23 void *calc_max(void *arg) {
24     max = numbers[0];
25     for(int i = 1; i < num_count; i++){
26
27         if(numbers[i] > max) {
28             max = numbers[i];
29         }
30     }
31     pthread_exit(NULL);
32 }
33
34
35
36
37 pthread_exit(NULL);
38 }
39 }
40
41 void *calc_min(void *arg) {
42     min = numbers[0];
43     for(int i = 1; i < num_count; i++) {
44         if(numbers[i] < min) {
45             min = numbers[i];
46         }
47     }
48     pthread_exit(NULL);
49 }
50
51 }
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78 }
79
80 rc = pthread_create(&threads[2], NULL, calc_min, NULL);
81 if(rc) {
82
83     printf("Error: Unable to create thread. \n");
84     exit(-1);
85
86 }
87
88 for(int i = 0; i < NUM_THREADS; i++) {
89
90     rc = pthread_join(threads[i], NULL);
91     if(rc) {
92
93         printf("Error: Unable to join thread. \n");
94         exit(-1);
95
96     }
97
98 }
99 printf("The average value is %.2f\n", average);
100 printf("The minimum value is %d\n", min);
101 printf("The maximum value is %d\n", max);
102 pthread_exit(NULL);
103
104 }
```

```
treydenwilson@treydenwilson-VirtualBox:~$ gedit
treydenwilson@treydenwilson-VirtualBox:~$ gcc Q2Lab4.c -o Q2Lab4
treydenwilson@treydenwilson-VirtualBox:~$ ./Q2Lab4
The average value is 44.29
The minimum value is 2
The maximum value is 98
treydenwilson@treydenwilson-VirtualBox:~$
```

The following code computes different statistical values of a set of numbers. One thread computes the greatest max value, another thread computes the min value, and a third thread computes the average. The program then prints out the average, minimum, and maximum value, which is 44.29, 2, and 98.

3. Write a C program that opens the file "outputLab4.txt" for writing and appends the phrase "This is a test for opening, writing, and closing a file!"

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <fcntl.h>
5
6 int main() {
7
8     int fd;
9     char buf[100] = "This is a test for opening, writing, and closing a file!";
10    ssize_t n;
11
12    fd = open("outputLab4.txt", O_WRONLY | O_CREAT, 0644);
13    if(fd == -1){
14
15        perror("open");
16        exit(EXIT_FAILURE);
17    }
18
19    n = write(fd, buf, sizeof(buf));
20    if(n == -1) {
21        perror("write");
22        exit(EXIT_FAILURE);
23    }
24
25    if(close(fd) == -1) {
26
27        perror("close");
28        exit(EXIT_FAILURE);
29    }
30 }
31
32 return 0;
33 |
34 }
```

```
treydenwilson@treydenwilson-VirtualBox:~$ gedit
treydenwilson@treydenwilson-VirtualBox:~$ gcc -g Q3Lab4.c -o Q3Lab4
treydenwilson@treydenwilson-VirtualBox:~$ ./Q3Lab4
treydenwilson@treydenwilson-VirtualBox:~$ cat outputLab4.txt
This is a test for opening, writing, and closing a file!treydenwilson@treydenwilson-VirtualBox:~$
```

The following code opens or creates the file outputLab4.txt which it appends the phrase “This is a test for opening, writing, and closing a file!”. In addition, the code take precautions and exits upon failures. Thus, it satisfies the requirements for question 3.

4. Write a program for matrix addition, subtraction and multiplication using multithreading.

```
1 #include <pthread.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 #define ROWS 3
6 #define COLS 3
7
8 int matrix_a[ROWS][COLS];
9 int matrix_b[ROWS][COLS];
10 int matrix_c[ROWS][COLS]; // to store the result of the operations
11 int operation_type; // 0 = addition, 1 = subtraction, 2 = multiplication
12
13 void *matrix_operation(void *arg) {
14     int row = *(int *) arg;
15     for (int j = 0; j < COLS; j++) {
16         if (operation_type == 0) {
17             matrix_c[row][j] = matrix_a[row][j] + matrix_b[row][j];
18         } else if (operation_type == 1) {
19             matrix_c[row][j] = matrix_a[row][j] - matrix_b[row][j];
20         } else if (operation_type == 2) {
21             int sum = 0;
22             for (int k = 0; k < ROWS; k++) {
23                 sum += matrix_a[row][k] * matrix_b[k][j];
24             }
25             matrix_c[row][j] = sum;
26         }
27     }
28     pthread_exit(NULL);
29 }
30
31 void print_matrix(int matrix[][COLS]) {
32     for (int i = 0; i < ROWS; i++) {
33         for (int j = 0; j < COLS; j++) {
34             printf("%d ", matrix[i][j]);
35         }
36         printf("\n");
37     }
38 }
39
40 int main() {
41     pthread_t threads[ROWS];
42
43     // initialize matrix_a and matrix_b with some values
44     for (int i = 0; i < ROWS; i++) {
45         for (int j = 0; j < COLS; j++) {
46             matrix_a[i][j] = i * COLS + j;
47             matrix_b[i][j] = j * ROWS + i;
48         }
49     }
50
51     // perform addition
52     operation_type = 0;
```

```

53     printf("Matrix addition:\n");
54     for (int i = 0; i < ROWS; i++) {
55         pthread_create(&threads[i], NULL, matrix_operation, (void *) &i);
56     }
57     for (int i = 0; i < ROWS; i++) {
58         pthread_join(threads[i], NULL);
59     }
60     print_matrix(matrix_c);
61
62     // perform subtraction
63     operation_type = 1;
64     printf("Matrix subtraction:\n");
65     for (int i = 0; i < ROWS; i++) {
66         pthread_create(&threads[i], NULL, matrix_operation, (void *) &i);
67     }
68     for (int i = 0; i < ROWS; i++) {
69         pthread_join(threads[i], NULL);
70     }
71     print_matrix(matrix_c);
72
73     // perform multiplication
74     operation_type = 2;
75     printf("Matrix multiplication:\n");
76
77     for (int i = 0; i < ROWS; i++) {
78         pthread_create(&threads[i], NULL, matrix_operation, (void *) &i);
79     }
80     for (int i = 0; i < ROWS; i++) {
81         pthread_join(threads[i], NULL);
82     }
83     print_matrix(matrix_c);
84
85     return 0;
86 }

treydenwilson@treydenwilson-VirtualBox:~$ gedit
treydenwilson@treydenwilson-VirtualBox:~$ gcc -g Q4Lab4.c -o Q4Lab4
treydenwilson@treydenwilson-VirtualBox:~$ ./Q4Lab4
Matrix addition:
0 0 0
0 0 0
0 0 0
Matrix subtraction:
0 0 0
0 0 0
0 0 0
Matrix multiplication:
0 0 0
0 0 0
0 0 0
treydenwilson@treydenwilson-VirtualBox:~$
```

The following program conducts matrix addition, subtraction and multiplication. In the screenshot above, zero matrices are used to easily identify that the program works since  $0+0=0$ ,  $0-0=0$ , and  $0*0=0$ . Furthermore, the matrix used is a 3x3 matrix and threads are implemented to compute.