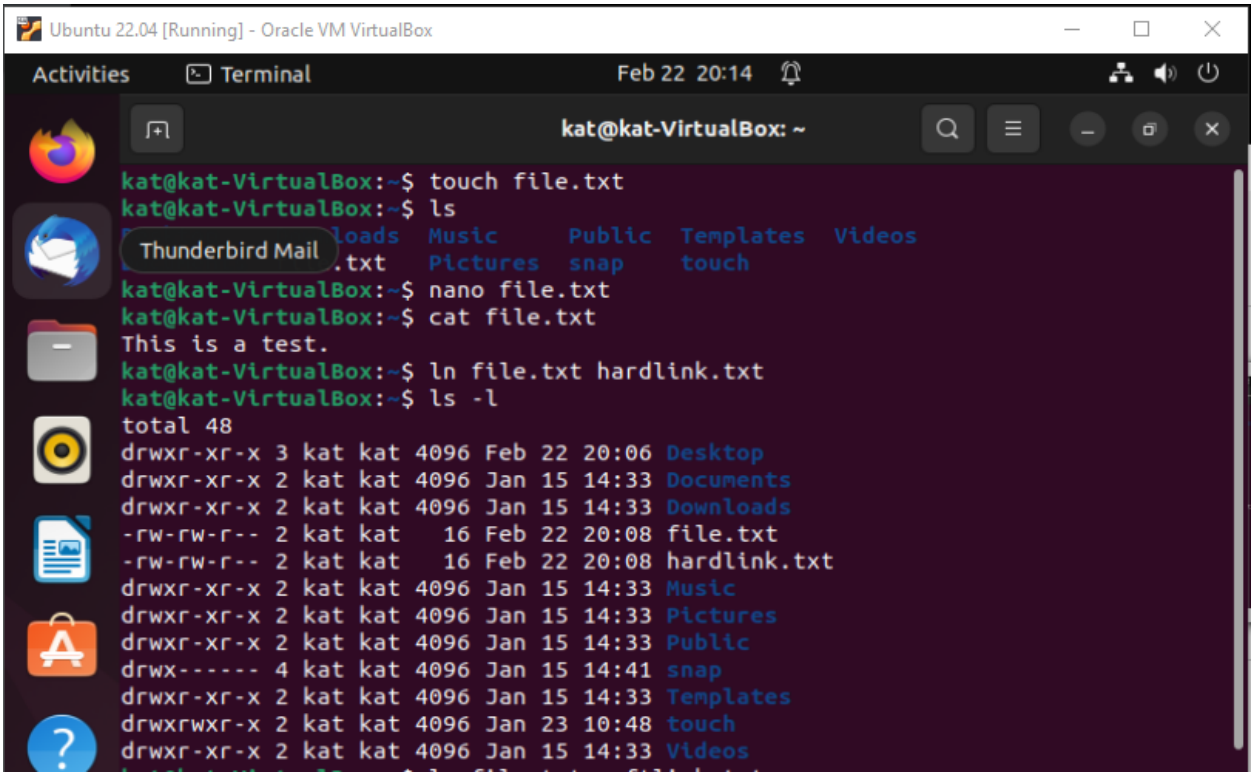
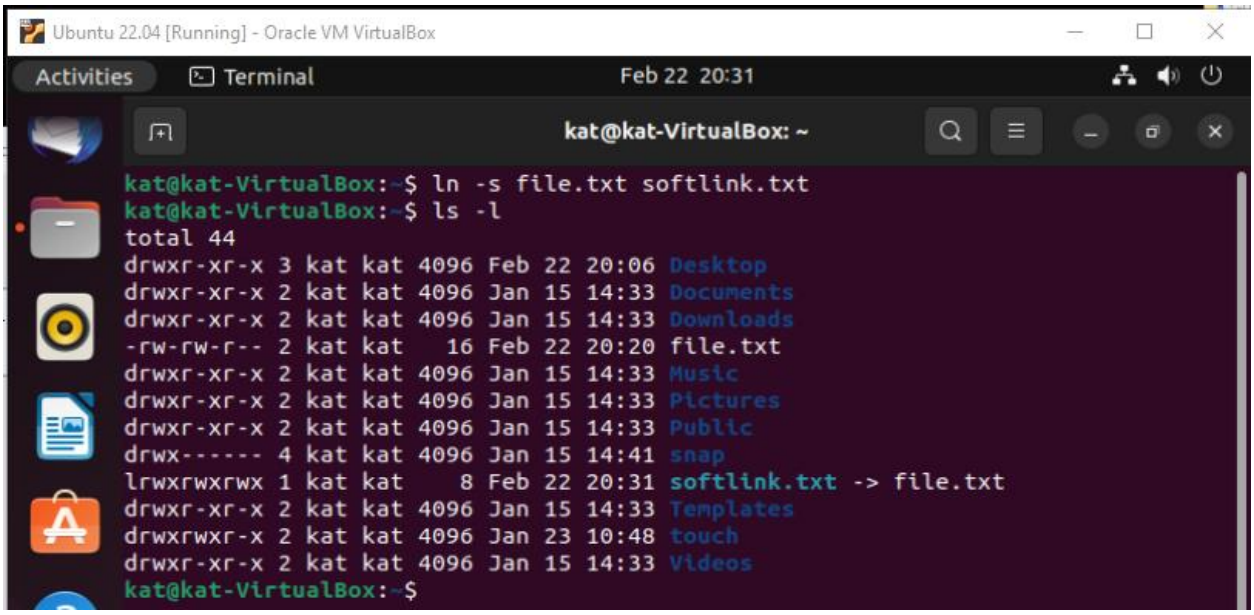
1.





When you create a hard link you create a new file that references or points to the exact spot on a hard drive where the Inodes stores the data. A soft link does not work the same way. A soft link isn't a separate file, it points to the same name of the original file, rather than to a spot on the hard drive.

2.

Ubuntu 22.04 [Running] - Oracle VM VirtualBox

Activities | Text Editor | Feb 22 20:47

Open | *lab4.c ~/Desktop | Save

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

#define NUM_THREADS 2

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

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);
}

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);
```

C | Tab Width: 8 | Ln 40, Col 4 | INS

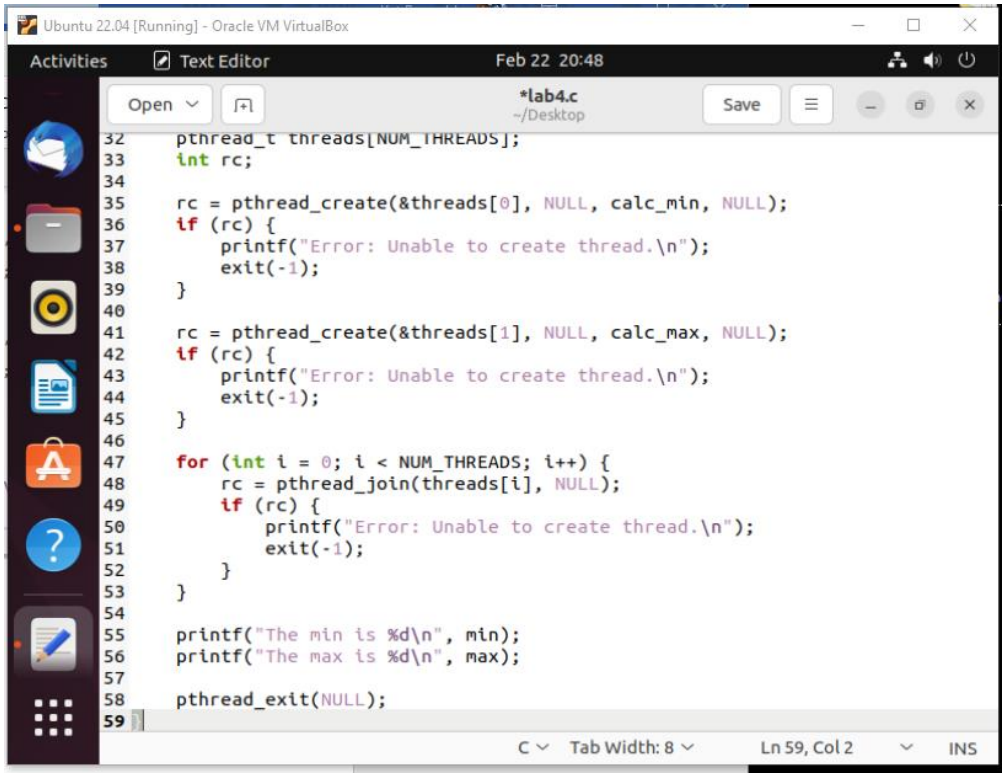
Ubuntu 22.04 [Running] - Oracle VM VirtualBox

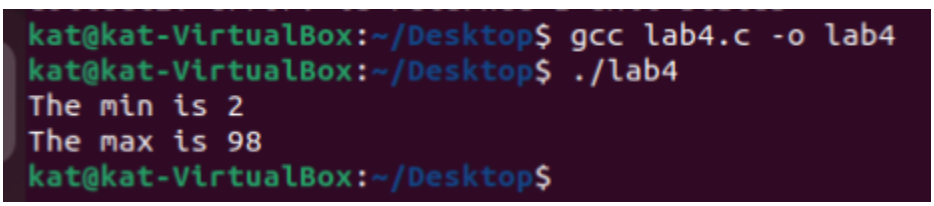
Activities | Text Editor | Feb 22 20:47

Open | *lab4.c ~/Desktop | Save

```c
            max = 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_min, 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);
    }

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

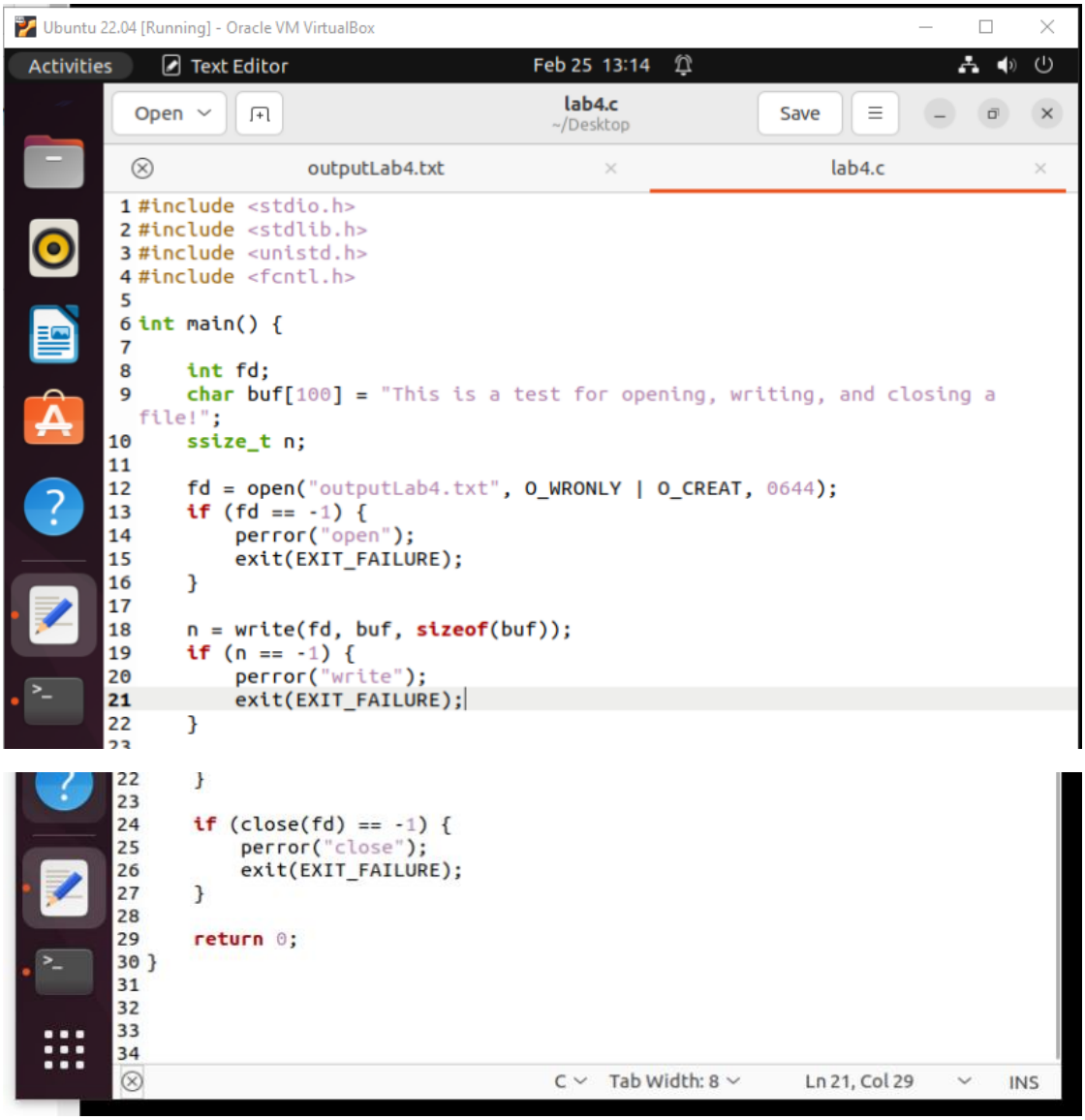
C | Tab Width: 8 | Ln 3, Col 20 | INS



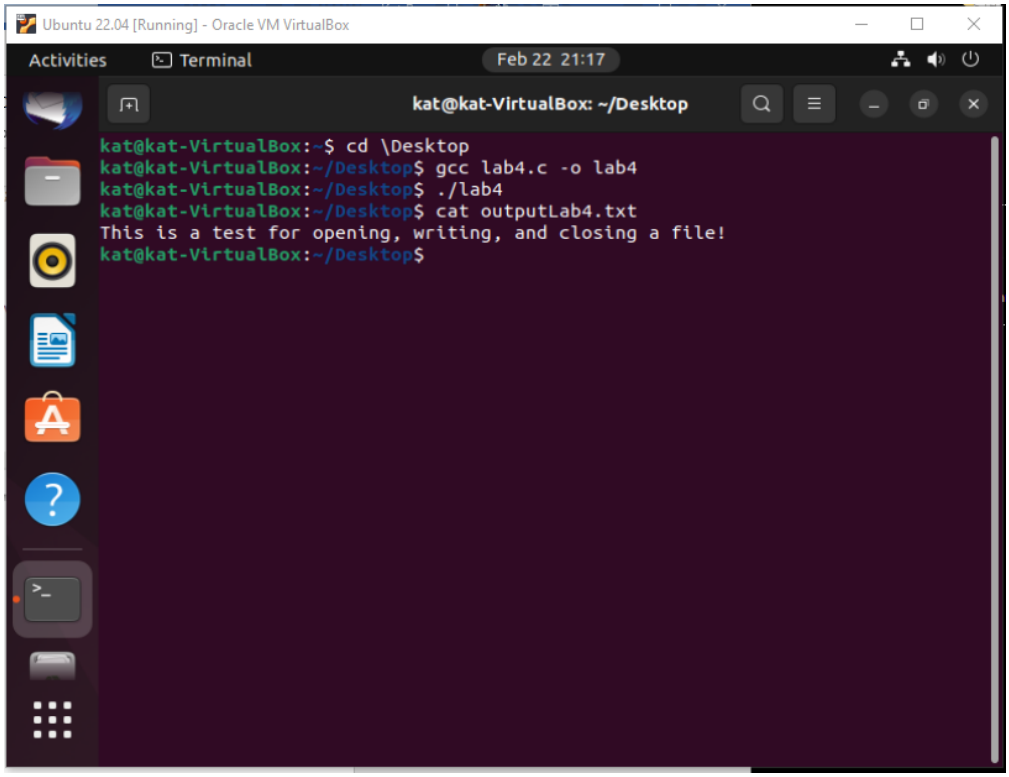


This code creates a multithreaded program that computes different the min and max of a given an array. There are three functions: a function finds the minimum value in a given array, a function to find the maximum value of a given array, and the main function. The main function is where the threads are created and executed. The pthread_t variable is need to store the thread ID, the pthread_create() function is used to pass the address of the pthread_t variable, the thread attributes, the function to run the new thread, and any arguments to pass that function.

3.


Ubuntu 22.04 [Running] - Oracle VM VirtualBox
Activities
Text Editor
Feb 25 13:14
Open
lab4.c
~/Desktop
Save
outputLab4.txt
lab4.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
int main() {
int fd;
char buf[100] = "This is a test for opening, writing, and closing a file!";
ssize_t n;
fd = open("outputLab4.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;
}
C
Tab Width: 8
Ln 21, Col 29
INS



The code uses the system calls for open and write. The open call opens the file and returns a file descriptor. The write call is used to write data to a file. The code opens the file “outputLab4.txt” and writes “This is a test for opening, writing, and closing a file!”.

4.

Ubuntu 22.04 [Running] - Oracle VM VirtualBox

Activities | Text Editor | Feb 24 14:53

Open | *lab4.c ~/Desktop | Save

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

#define NUM_THREADS 3
#define R1 2
#define C1 2
#define R2 2
#define C2 2

int mat1[R1][C1] = { {5, 6}, {1, 3} };
int mat2[R2][C2] = { {2, 5}, {9, 4} };

int matA[R1][C2];
int matS[R1][C2];
int matM[R1][C2];
```

```c

void *matrix_add(void *arg) {

    for (int i = 0; i < R1; i++) {
        for (int j = 0; j < C2; j++) {
            matA[i][j] = 0;
            matA[i][j] += mat1[i][j] + mat2[i][j];
        }
    }
    printf("Matrix addition results:\n");
    for (int i = 0; i < R1; i++) {
        for (int j = 0; j < C2; j++) {
            printf(" %d ", matA[i][j]);
        }
        printf("\n");
    }
    pthread_exit(NULL);
}
```

```c

void *matrix_sub(void *arg) {

    for (int i = 0; i < R1; i++) {
        for (int j = 0; j < C2; j++) {
            matS[i][j] = 0;
            matS[i][j] += mat1[i][j] - mat2[i][j];
        }
    }
    printf("Matrix subtraction results:\n");
        for (int i = 0; i < R1; i++) {
            for (int j = 0; j < C2; j++) {
                 printf(" %d ", matS[i][j]);
        }
        printf("\n");
    }
    pthread_exit(NULL);
}

```

```c

void *matrix_mult(void *arg) {

    for (int i = 0; i < R1; i++) {
        for (int j = 0; j < C2; j++) {
            matM[i][j] = 0;
            for (int k = 0; k < R2; k++) {
                matM[i][j] += mat1[i][j] * mat2[i][j];
            }
        }
    }
    printf("Matrix multiplication results:\n");
    for (int i = 0; i < R1; i++) {
        for (int j = 0; j < C2; j++) {
            printf(" %d ", matM[i][j]);
        }
        printf("\n");
    }
    pthread_exit(NULL);
}

```

Ubuntu 22.04 [Running] - Oracle VM VirtualBox

Activities | Text Editor | Feb 24 14:54

Open | *lab4.c ~/Desktop | Save

```c

int main(int argc, char *argv[]) {

    pthread_t threads[NUM_THREADS];
    int rc;

    rc = pthread_create(&threads[0], NULL, matrix_add, NULL);
    if (rc) {
        printf("Error: Unable to create thread.\n");
        exit(-1);
    }

    rc = pthread_create(&threads[1], NULL, matrix_sub, NULL);
    if (rc) {
        printf("Error: Unable to create thread.\n");
        exit(-1);
    }

    rc = pthread_create(&threads[2], NULL, matrix_mult, 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);
        }
    }

    pthread_exit(NULL);
}


```

C | Tab Width: 8 | Ln 70, Col 6 | INS

```
kat@kat-VirtualBox:~/Desktop$ gcc lab4.c -o lab4
kat@kat-VirtualBox:~/Desktop$ ./lab4
Matrix addition results:
 7  11
 10  7
Matrix subtraction results:
 3  1
 -8  -1
Matrix multiplication results:
 20  60
 18  24
kat@kat-VirtualBox:~/Desktop$
```

This code creates a multithreaded program that computes different the min and max of a given an array. There are four functions: a function find the minimum value in a given array, a function to find the maximum value of a given array, and the main function. The main function is where the threads are created and executed. The pthread_t variable is need to store the thread ID, the pthread_create() function is used to pass the address of the pthread_t variable, the thread attributes, the function to run the new thread, and any arguments to pass that function.