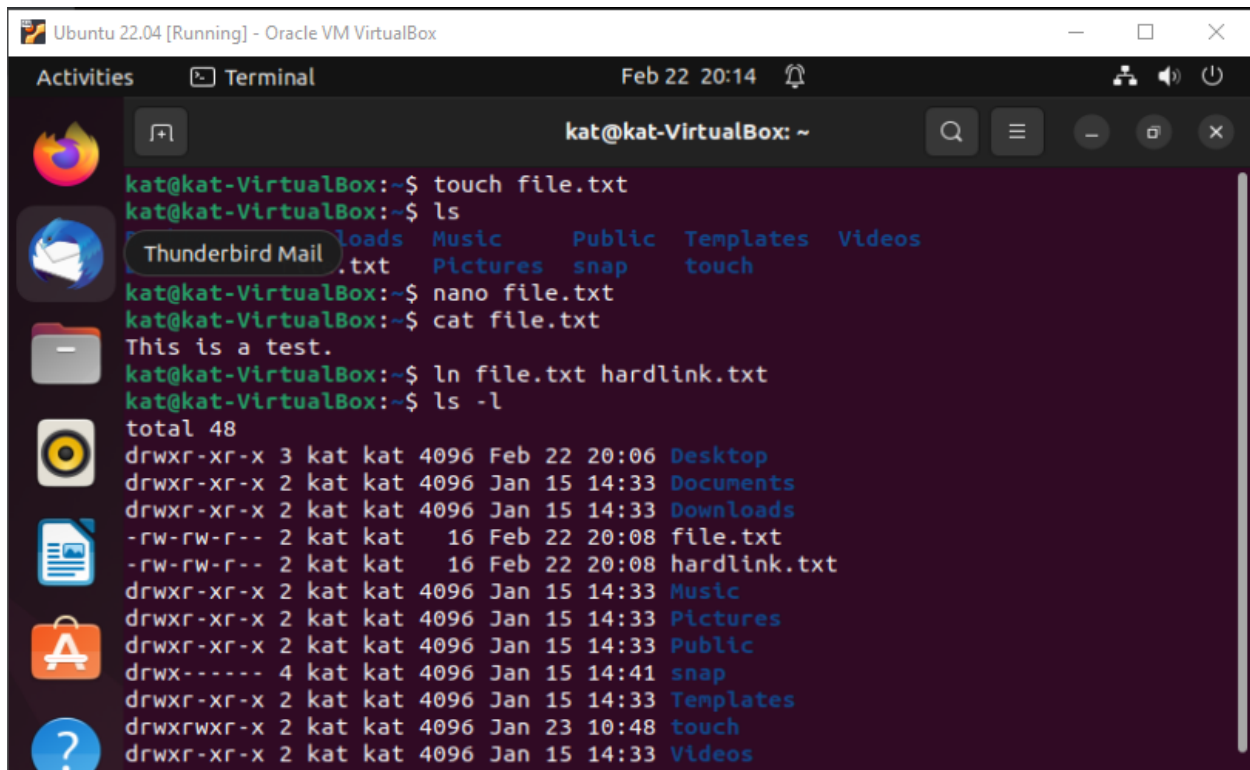
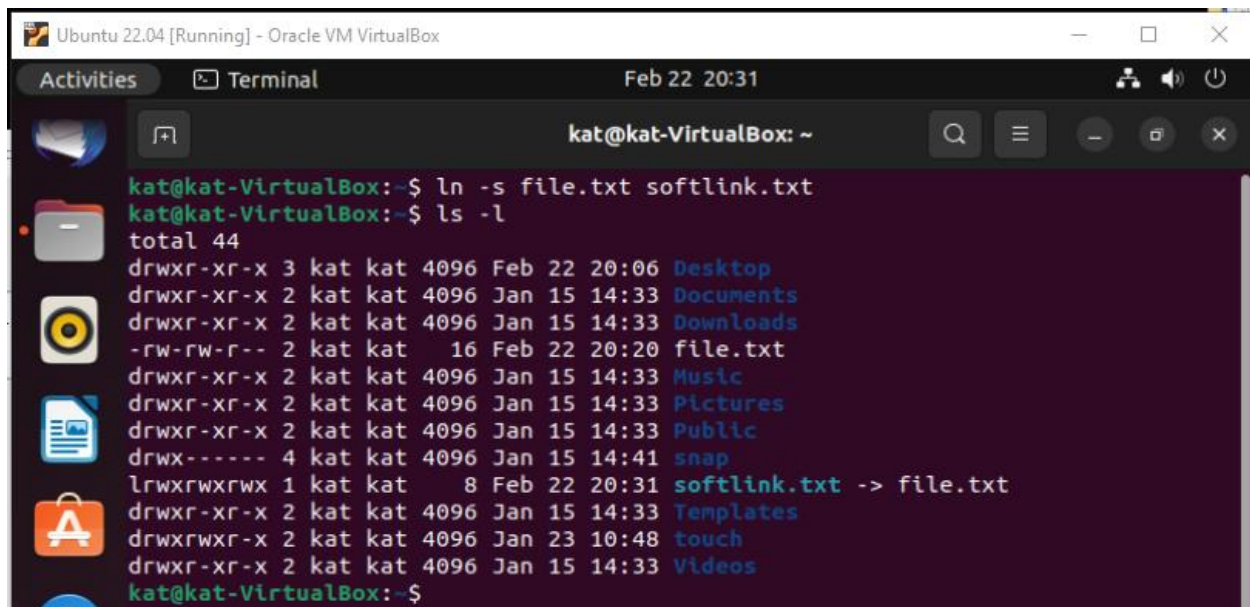


1.



A terminal window titled 'Ubuntu 22.04 [Running] - Oracle VM VirtualBox' showing a series of commands and their outputs. The user 'kat' is at the prompt. The commands and outputs are as follows:

```
kat@kat-VirtualBox:~$ touch file.txt
kat@kat-VirtualBox:~$ ls
Desktop  Downloads  file.txt  Music  Public  Templates  Videos
kat@kat-VirtualBox:~$ nano file.txt
kat@kat-VirtualBox:~$ cat file.txt
This is a test.
kat@kat-VirtualBox:~$ ln file.txt hardlink.txt
kat@kat-VirtualBox:~$ ls -l
total 48
drwxr-xr-x 3 kat kat 4096 Feb 22 20:06 Desktop
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Documents
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Downloads
-rw-rw-r-- 2 kat kat 16 Feb 22 20:08 file.txt
-rw-rw-r-- 2 kat kat 16 Feb 22 20:08 hardlink.txt
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Music
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Pictures
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Public
drwx----- 4 kat kat 4096 Jan 15 14:41 snap
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Templates
drwxrwxr-x 2 kat kat 4096 Jan 23 10:48 touch
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Videos
```



A terminal window titled 'Ubuntu 22.04 [Running] - Oracle VM VirtualBox' showing a series of commands and their outputs. The user 'kat' is at the prompt. The commands and outputs are as follows:

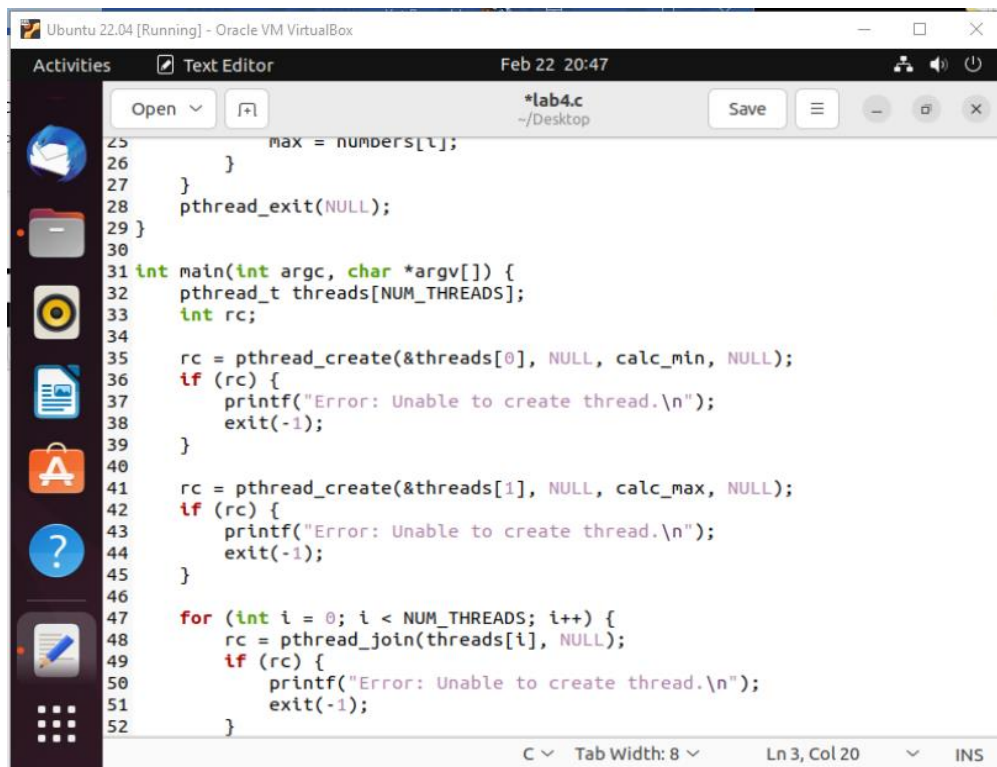
```
kat@kat-VirtualBox:~$ ln -s file.txt softlink.txt
kat@kat-VirtualBox:~$ ls -l
total 44
drwxr-xr-x 3 kat kat 4096 Feb 22 20:06 Desktop
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Documents
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Downloads
-rw-rw-r-- 2 kat kat 16 Feb 22 20:20 file.txt
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Music
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Pictures
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Public
drwx----- 4 kat kat 4096 Jan 15 14:41 snap
lrwxrwxrwx 1 kat kat 8 Feb 22 20:31 softlink.txt -> file.txt
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Templates
drwxrwxr-x 2 kat kat 4096 Jan 23 10:48 touch
drwxr-xr-x 2 kat kat 4096 Jan 15 14:33 Videos
kat@kat-VirtualBox:~$
```

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.



```
1 #include <pthread.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 #define NUM_THREADS 2
6
7 int numbers[] = {2, 20, 25, 5, 70, 90, 98};
8 int num_count = sizeof(numbers) / sizeof(int);
9 int max, min;
10
11 void *calc_min(void *arg) {
12     min = numbers[0];
13     for (int i = 1; i < num_count; i++) {
14         if (numbers[i] < min) {
15             min = numbers[i];
16         }
17     }
18     pthread_exit(NULL);
19 }
20
21 void *calc_max(void *arg) {
22     max = numbers[0];
23     for (int i = 1; i < num_count; i++) {
24         if (numbers[i] > max) {
25             max = numbers[i];
26         }
27     }
28     pthread_exit(NULL);
29 }
```



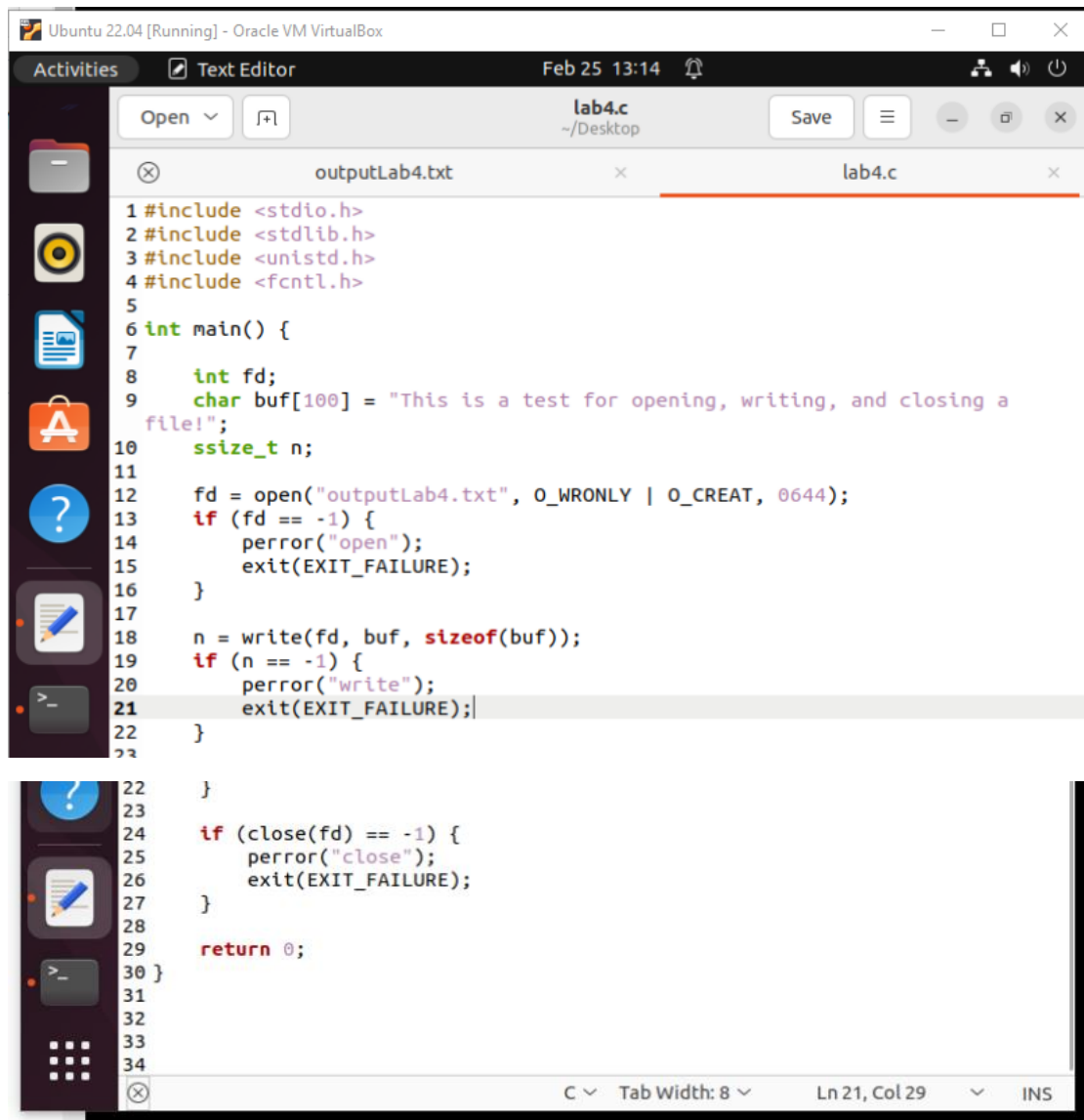
```
25     max = numbers[i];
26 }
27 }
28 pthread_exit(NULL);
29 }
30
31 int main(int argc, char *argv[]) {
32     pthread_t threads[NUM_THREADS];
33     int rc;
34
35     rc = pthread_create(&threads[0], NULL, calc_min, NULL);
36     if (rc) {
37         printf("Error: Unable to create thread.\n");
38         exit(-1);
39     }
40
41     rc = pthread_create(&threads[1], NULL, calc_max, NULL);
42     if (rc) {
43         printf("Error: Unable to create thread.\n");
44         exit(-1);
45     }
46
47     for (int i = 0; i < NUM_THREADS; i++) {
48         rc = pthread_join(threads[i], NULL);
49         if (rc) {
50             printf("Error: Unable to create thread.\n");
51             exit(-1);
52         }
53     }
54 }
```

```
32 pthread_t threads[NUM_THREADS];
33 int rc;
34
35 rc = pthread_create(&threads[0], NULL, calc_min, NULL);
36 if (rc) {
37     printf("Error: Unable to create thread.\n");
38     exit(-1);
39 }
40
41 rc = pthread_create(&threads[1], NULL, calc_max, NULL);
42 if (rc) {
43     printf("Error: Unable to create thread.\n");
44     exit(-1);
45 }
46
47 for (int i = 0; i < NUM_THREADS; i++) {
48     rc = pthread_join(threads[i], NULL);
49     if (rc) {
50         printf("Error: Unable to create thread.\n");
51         exit(-1);
52     }
53 }
54
55 printf("The min is %d\n", min);
56 printf("The max is %d\n", max);
57
58 pthread_exit(NULL);
59
```

```
kat@kat-VirtualBox:~/Desktop$ gcc lab4.c -o lab4
kat@kat-VirtualBox:~/Desktop$ ./lab4
The min is 2
The max is 98
kat@kat-VirtualBox:~/Desktop$
```

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.

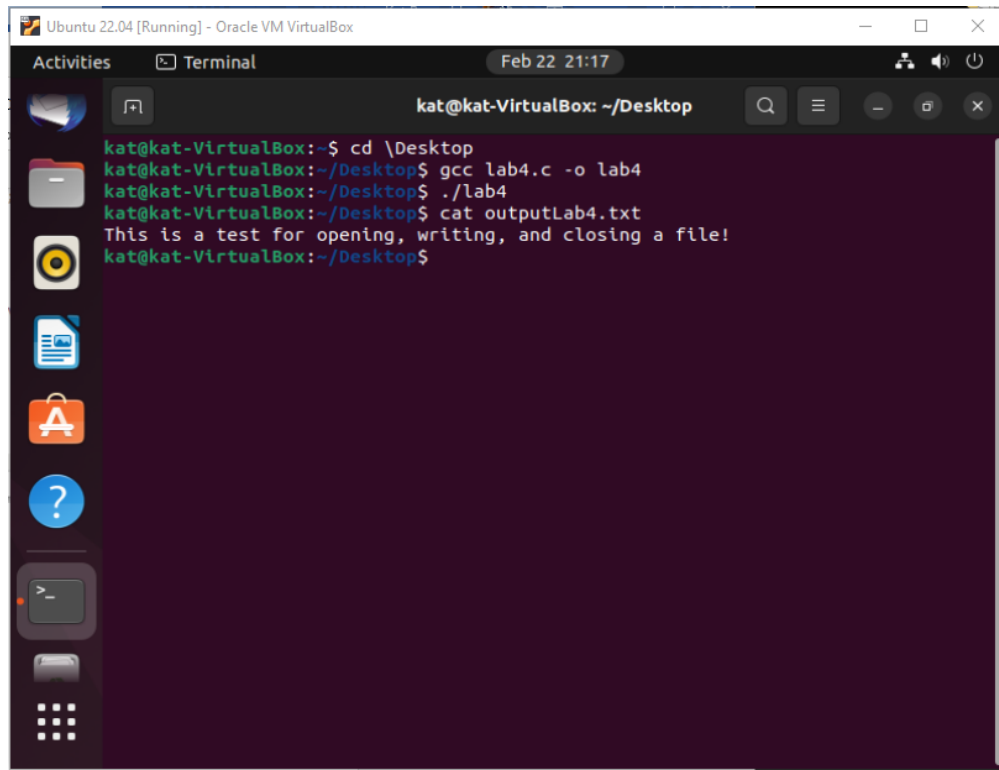
3.



The screenshot shows a text editor window titled "lab4.c" with a file path of "~/Desktop". The editor contains a C program that opens a file, writes a message, and closes it. The code is as follows:

```
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        perror("open");
15        exit(EXIT_FAILURE);
16    }
17
18    n = write(fd, buf, sizeof(buf));
19    if (n == -1) {
20        perror("write");
21        exit(EXIT_FAILURE);
22    }
23
24    if (close(fd) == -1) {
25        perror("close");
26        exit(EXIT_FAILURE);
27    }
28
29    return 0;
30 }
```

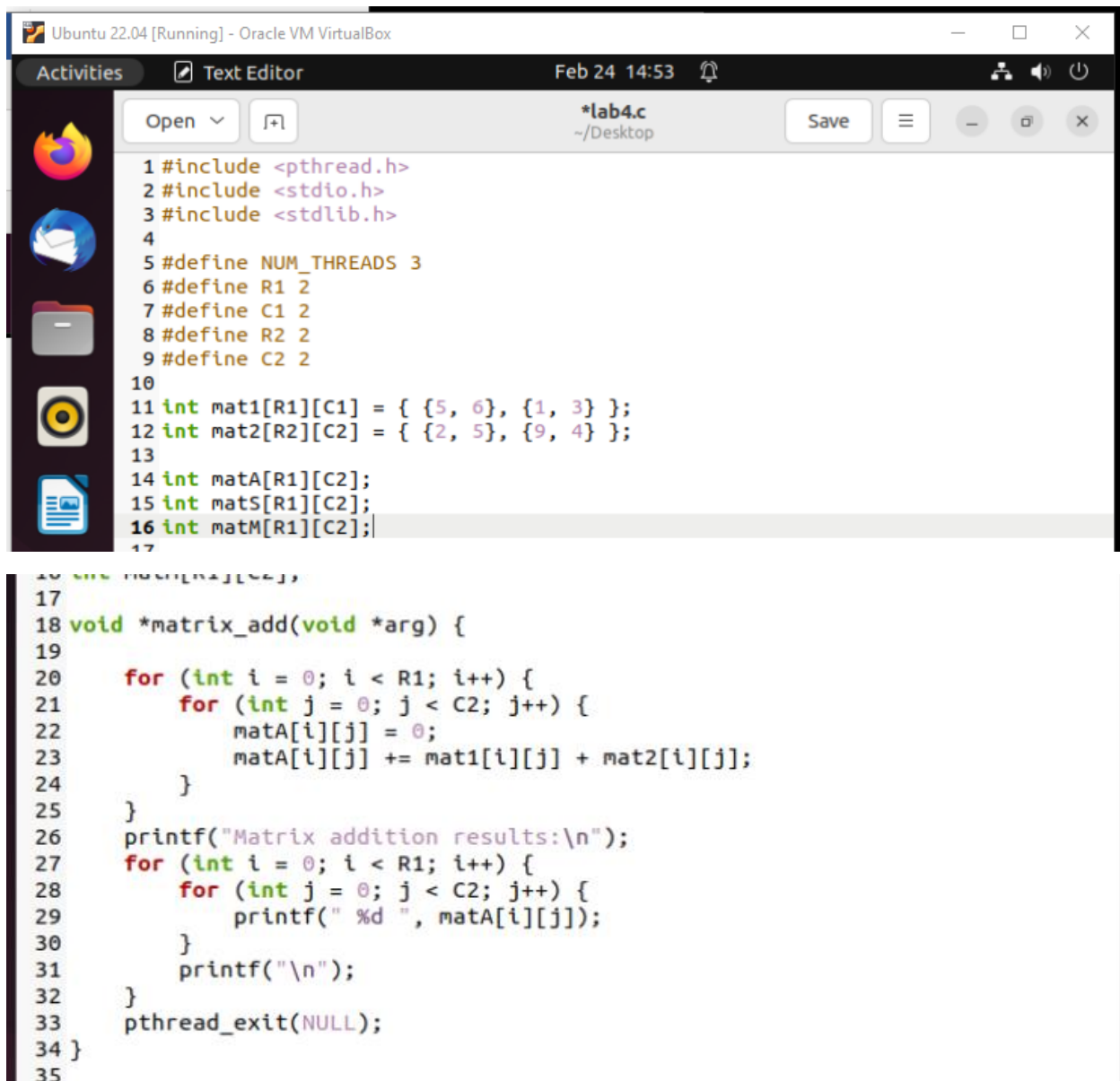
The status bar at the bottom indicates "C", "Tab Width: 8", "Ln 21, Col 29", and "INS".



```
kat@kat-VirtualBox: ~/$ cd \Desktop
kat@kat-VirtualBox:~/Desktop$ gcc lab4.c -o lab4
kat@kat-VirtualBox:~/Desktop$ ./lab4
kat@kat-VirtualBox:~/Desktop$ cat outputLab4.txt
This is a test for opening, writing, and closing a file!
kat@kat-VirtualBox:~/Desktop$
```

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.



The screenshot shows a text editor window titled "Ubuntu 22.04 [Running] - Oracle VM VirtualBox" with a dark theme. The editor is open to a file named "lab4.c" located at "~/Desktop". The code is a C program that demonstrates matrix addition using pthreads. It includes headers for pthread.h, stdio.h, and stdlib.h. It defines constants for the number of threads (3), rows (2), and columns (2). It declares three 2D arrays: mat1, mat2, and matA. The main function is not shown, but the code defines a matrix_add function that takes a void pointer argument. Inside matrix_add, it initializes matA to zero and then adds the elements of mat1 and mat2 to matA. It prints the results of the addition and then exits the thread.

```
1 #include <pthread.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 #define NUM_THREADS 3
6 #define R1 2
7 #define C1 2
8 #define R2 2
9 #define C2 2
10
11 int mat1[R1][C1] = { {5, 6}, {1, 3} };
12 int mat2[R2][C2] = { {2, 5}, {9, 4} };
13
14 int matA[R1][C2];
15 int mats[R1][C2];
16 int matM[R1][C2];
17
18 void *matrix_add(void *arg) {
19     for (int i = 0; i < R1; i++) {
20         for (int j = 0; j < C2; j++) {
21             matA[i][j] = 0;
22             matA[i][j] += mat1[i][j] + mat2[i][j];
23         }
24     }
25     printf("Matrix addition results:\n");
26     for (int i = 0; i < R1; i++) {
27         for (int j = 0; j < C2; j++) {
28             printf(" %d ", matA[i][j]);
29         }
30         printf("\n");
31     }
32     pthread_exit(NULL);
33 }
34 }
35
```

```

35
36 void *matrix_sub(void *arg) {
37     for (int i = 0; i < R1; i++) {
38         for (int j = 0; j < C2; j++) {
39             matS[i][j] = 0;
40             matS[i][j] += mat1[i][j] - mat2[i][j];
41         }
42     }
43     printf("Matrix subtraction results:\n");
44     for (int i = 0; i < R1; i++) {
45         for (int j = 0; j < C2; j++) {
46             printf(" %d ", matS[i][j]);
47         }
48         printf("\n");
49     }
50     pthread_exit(NULL);
51 }
52 }
53
54 void *matrix_mult(void *arg) {
55     for (int i = 0; i < R1; i++) {
56         for (int j = 0; j < C2; j++) {
57             matM[i][j] = 0;
58             for (int k = 0; k < R2; k++) {
59                 matM[i][j] += mat1[i][k] * mat2[k][j];
60             }
61         }
62     }
63     printf("Matrix multiplication results:\n");
64     for (int i = 0; i < R1; i++) {
65         for (int j = 0; j < C2; j++) {
66             printf(" %d ", matM[i][j]);
67         }
68         printf("\n");
69     }
70     pthread_exit(NULL);
71 }
72
73
74

```


Ubuntu 22.04 [Running] - Oracle VM VirtualBox

Activities Text Editor Feb 24 14:54

*lab4.c
~/Desktop

Open Save

```
74
75 int main(int argc, char *argv[]) {
76     pthread_t threads[NUM_THREADS];
77     int rc;
78
79     rc = pthread_create(&threads[0], NULL, matrix_add, NULL);
80     if (rc) {
81         printf("Error: Unable to create thread.\n");
82         exit(-1);
83     }
84
85     rc = pthread_create(&threads[1], NULL, matrix_sub, NULL);
86     if (rc) {
87         printf("Error: Unable to create thread.\n");
88         exit(-1);
89     }
90
91     rc = pthread_create(&threads[2], NULL, matrix_mult, NULL);
92     if (rc) {
93         printf("Error: Unable to create thread.\n");
94         exit(-1);
95     }
96
97     for (int i = 0; i < NUM_THREADS; i++) {
98         rc = pthread_join(threads[i], NULL);
99         if (rc) {
100             printf("Error: Unable to join thread.\n");
101             exit(-1);
102         }
103     }
104
105     pthread_exit(NULL);
106 }
107
108
109
```

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.