11) Implementing ftruncate()

- In C, the ftruncate() function allows you to change the size of an open file. It comes in handy when you want to truncate or extend the file to a desired specific length. It belongs to the unistd.h library and works with file descriptors. When ftruncate(fd, size) is called, it instructs the system to truncate the file pointed to by the file descriptor fd to the new length size (in bytes). If the file was bigger, delete the rest; if smaller, the system would normally fill the extra space with null bytes (although this never happens in practice). Uses of ftruncate:

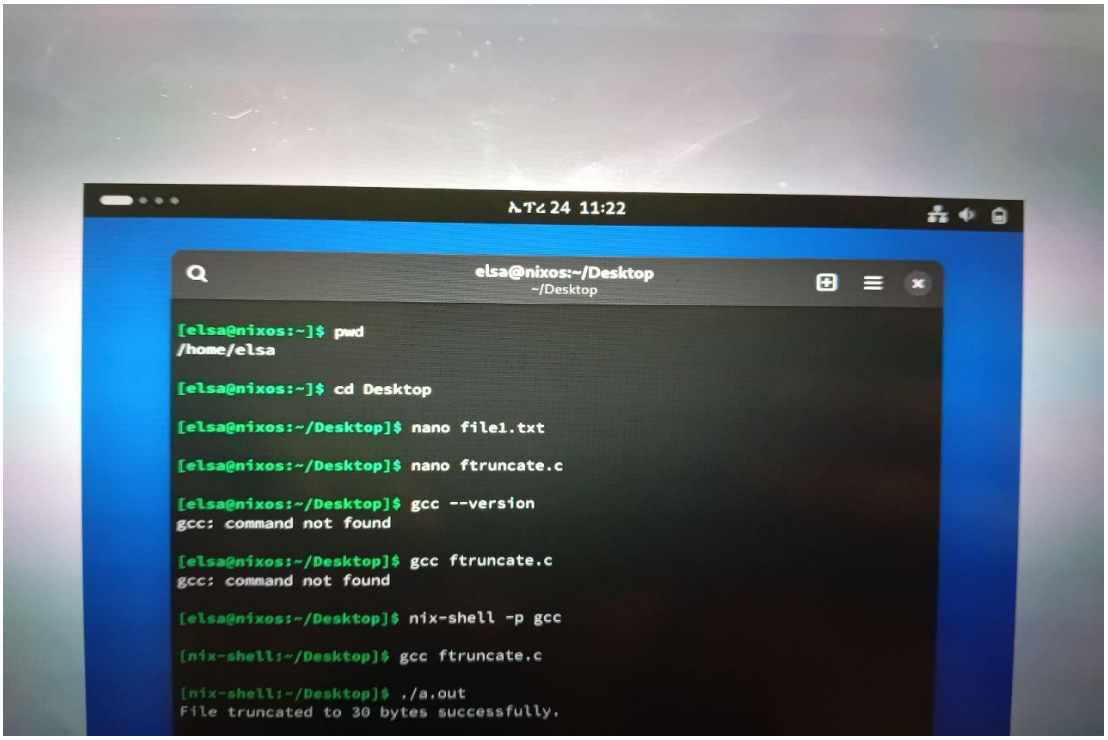
- To clean temporary files by strictly cutting the useful part.
- To reset or shrink logs, databases, or any kind of file.
- Useful in low-level system programming to get manual control over file size.

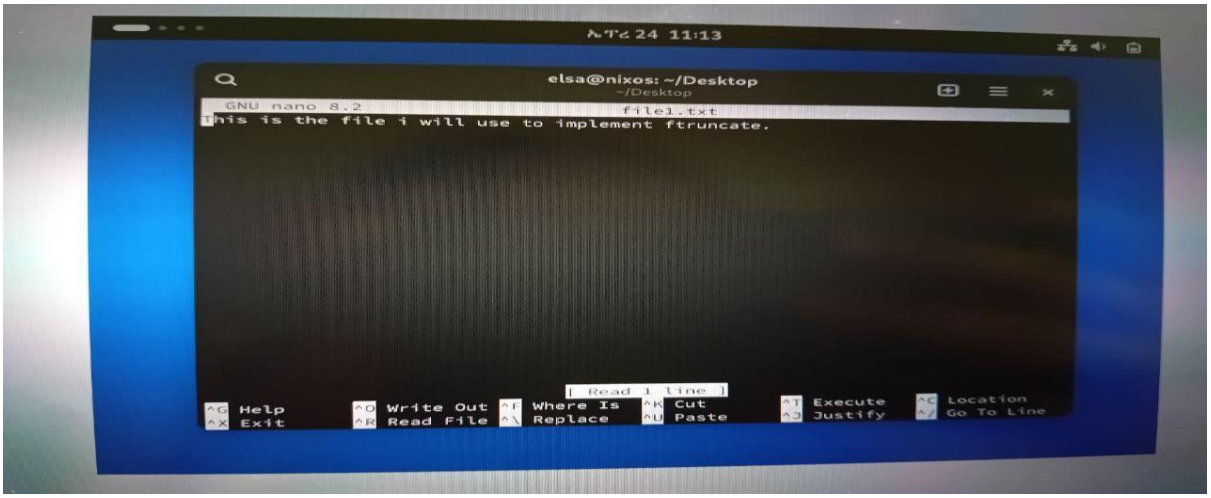
- Example :

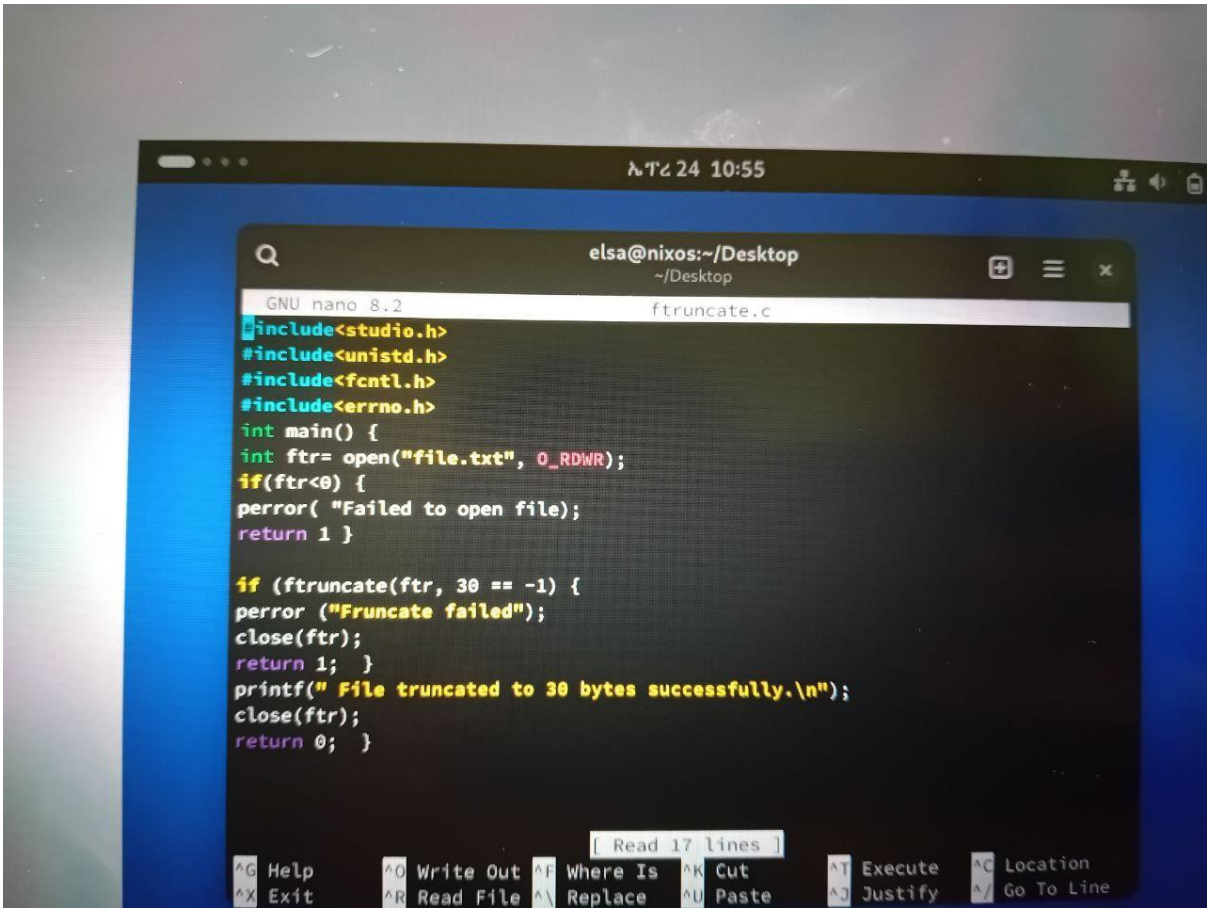
```
int ftr=open("file1.txt",O_RDWR);

ftruncate(ftr,30);
```

- Thus file "file1.txt" is opened and reshaped to 20 bytes:
- This means: "Keep the first 30 bytes of this file, and drop the rest."
- Here are the snipped images when I implement ftruncate() :




elsa@nixos: ~/Desktop
~/Desktop
GNU nano 8.2
file1.txt
This is the file i will use to implement ftruncate.
[Read 1 line]
^G Help
^X Exit
^O Write Out
^R Read File
^F Where Is
^\ Replace
^K Cut
^U Paste
^T Execute
^J Justify
^C Location
^/ Go To Line


elsa@nixos:~/Desktop
~/Desktop
GNU nano 8.2
ftruncate.c
#include<studio.h>
#include<unistd.h>
#include<fcntl.h>
#include<errno.h>
int main() {
int ftr= open("file.txt", O_RDWR);
if(ftr<0) {
perror("Failed to open file);
return 1 }
if (ftruncate(ftr, 30 == -1) {
perror ("Fruncate failed");
close(ftr);
return 1; }
printf(" File truncated to 30 bytes successfully.\n");
close(ftr);
return 0; }
[Read 17 lines]
^G Help
^X Exit
^O Write Out
^R Read File
^F Where Is
^\ Replace
^K Cut
^U Paste
^T Execute
^J Justify
^C Location
^/ Go To Line