

LAPORAN PRAKTIKUM SISTEM OPERASI



Disusun oleh :

FAIZAL AHMAD DENA L200210264

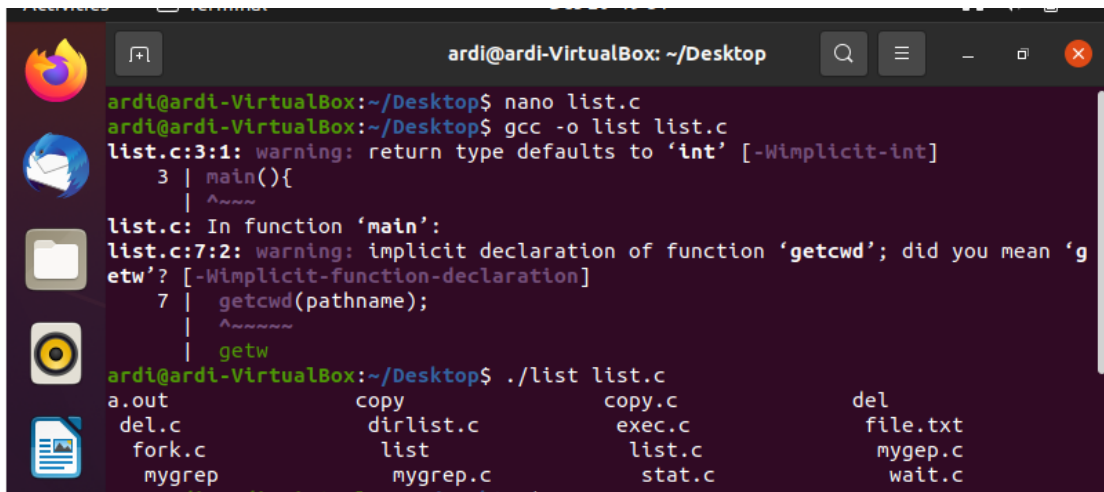
**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS KOMUNIKASI DAN INFORMATIKA
UNIVERSITAS MUHAMMADIYAH SURAKARTA
TAHUN 2021/2022**

Lembar Kerja Modul 10

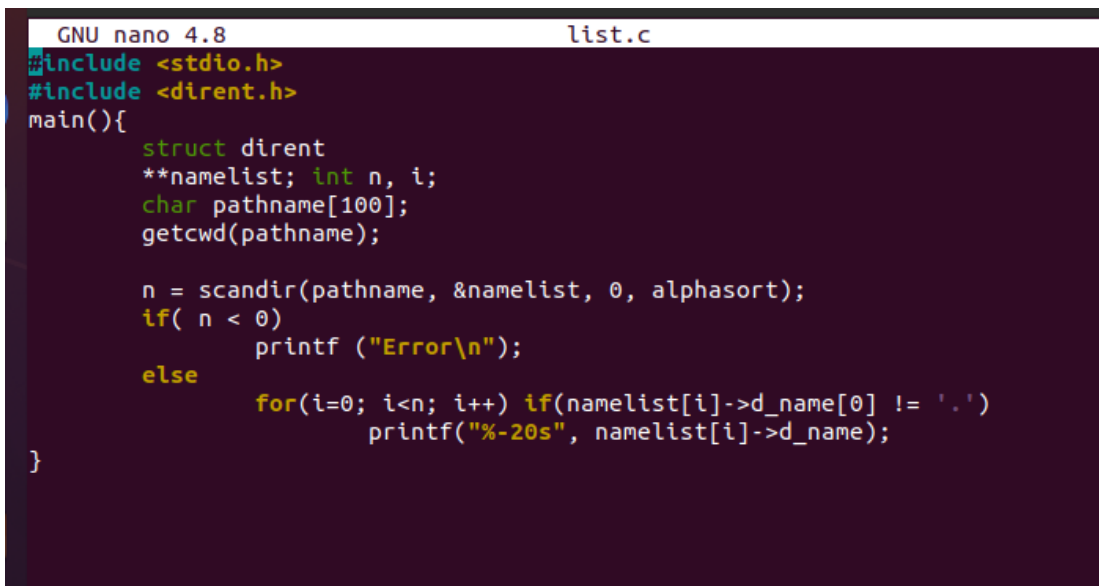
Nama : Faizal Ahmad Dena	Nilai Praktek :
NIM : L200210264	
Nama Asisten :	Tanda Tangan :
Tanggal Praktikum : -	

TUGAS

1. List.c untuk menampilkan file yang ada



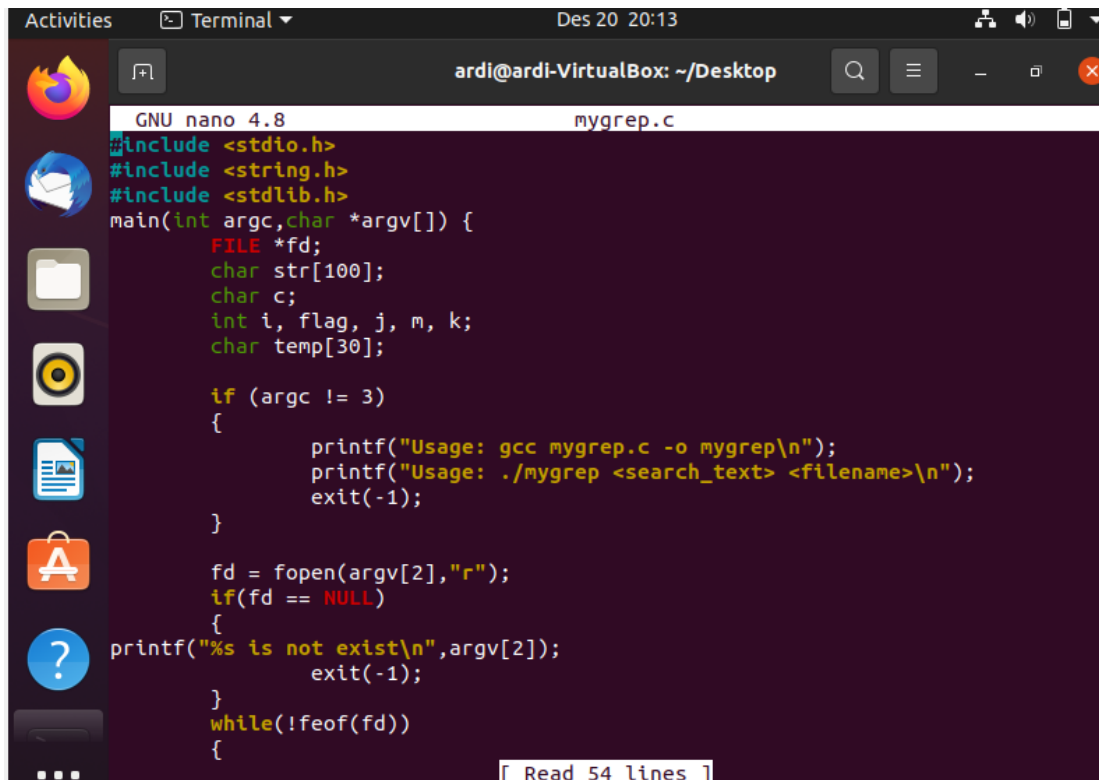
```
ardi@ardi-VirtualBox: ~/Desktop
ardi@ardi-VirtualBox:~/Desktop$ nano list.c
ardi@ardi-VirtualBox:~/Desktop$ gcc -o list list.c
list.c:3:1: warning: return type defaults to 'int' [-Wimplicit-int]
    3 | main(){
      | ^~~~~~
list.c: In function 'main':
list.c:7:2: warning: implicit declaration of function 'getcwd'; did you mean 'getw'? [-Wimplicit-function-declaration]
    7 |     cwd = getcwd(pathname);
      |           ^~~~~~
ardi@ardi-VirtualBox:~/Desktop$ ./list list.c
a.out          copy          copy.c          del
del.c          dirlist.c     exec.c          file.txt
fork.c         list             list.c          mygep.c
mygrep         mygrep.c         stat.c          wait.c
```



```
GNU nano 4.8 list.c
#include <stdio.h>
#include <dirent.h>
main(){
    struct dirent
    **namelist; int n, i;
    char pathname[100];
    getcwd(pathname);

    n = scandir(pathname, &namelist, 0, alphasort);
    if( n < 0)
        printf ("Error\n");
    else
        for(i=0; i<n; i++) if(namelist[i]->d_name[0] != '.')
            printf("%-20s", namelist[i]->d_name);
}
```

2. Mygrep.c untuk menampilkan teks atau kata yang ingin dicari dari file.txt



The screenshot shows a terminal window titled "ardl@ardl-VirtualBox: ~/Desktop" with a nano 4.8 editor open to the file "mygrep.c". The code visible includes headers, variable declarations, and the start of the main function with argument handling.

```
GNU nano 4.8 mygrep.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
main(int argc, char *argv[]) {
    FILE *fd;
    char str[100];
    char c;
    int i, flag, j, m, k;
    char temp[30];

    if (argc != 3)
    {
        printf("Usage: gcc mygrep.c -o mygrep\n");
        printf("Usage: ./mygrep <search_text> <filename>\n");
        exit(-1);
    }

    fd = fopen(argv[2], "r");
    if (fd == NULL)
    {
        printf("%s is not exist\n", argv[2]);
        exit(-1);
    }
    while (!feof(fd))
    {
```

[Read 54 lines]

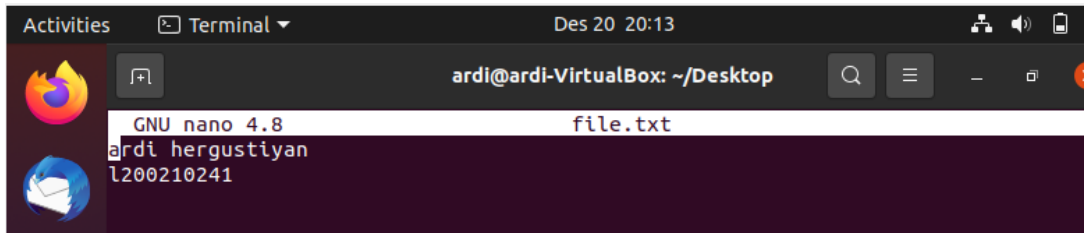


The screenshot shows the continuation of the mygrep.c code in the nano editor. It includes the logic for reading characters from the file, checking for the search string, and printing matches.

```
        i = 0;
        while (1)
        {
            c = fgetc(fd);
            if (feof(fd))
            {
                str[i++] = '\0'; break;
            }
            if (c == '\n')
            {
                str[i++] = '\0'; break;
            }
            str[i++] = c;
        }

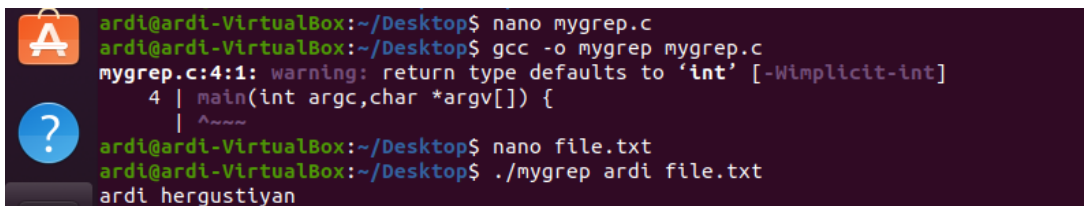
        if (strlen(str) >= strlen(argv[1]))
        for (k=0; k<=strlen(str)-strlen(argv[1]); k++)
        {
            for (m=0; m<strlen(argv[1]); m++)
                temp[m] = str[k+m];
            temp[m] = '\0';
            if (strcmp(temp, argv[1]) == 0)
            {
                printf("%s\n", str);
                break;
            }
        }
    }
}
```

Code mygrep.c



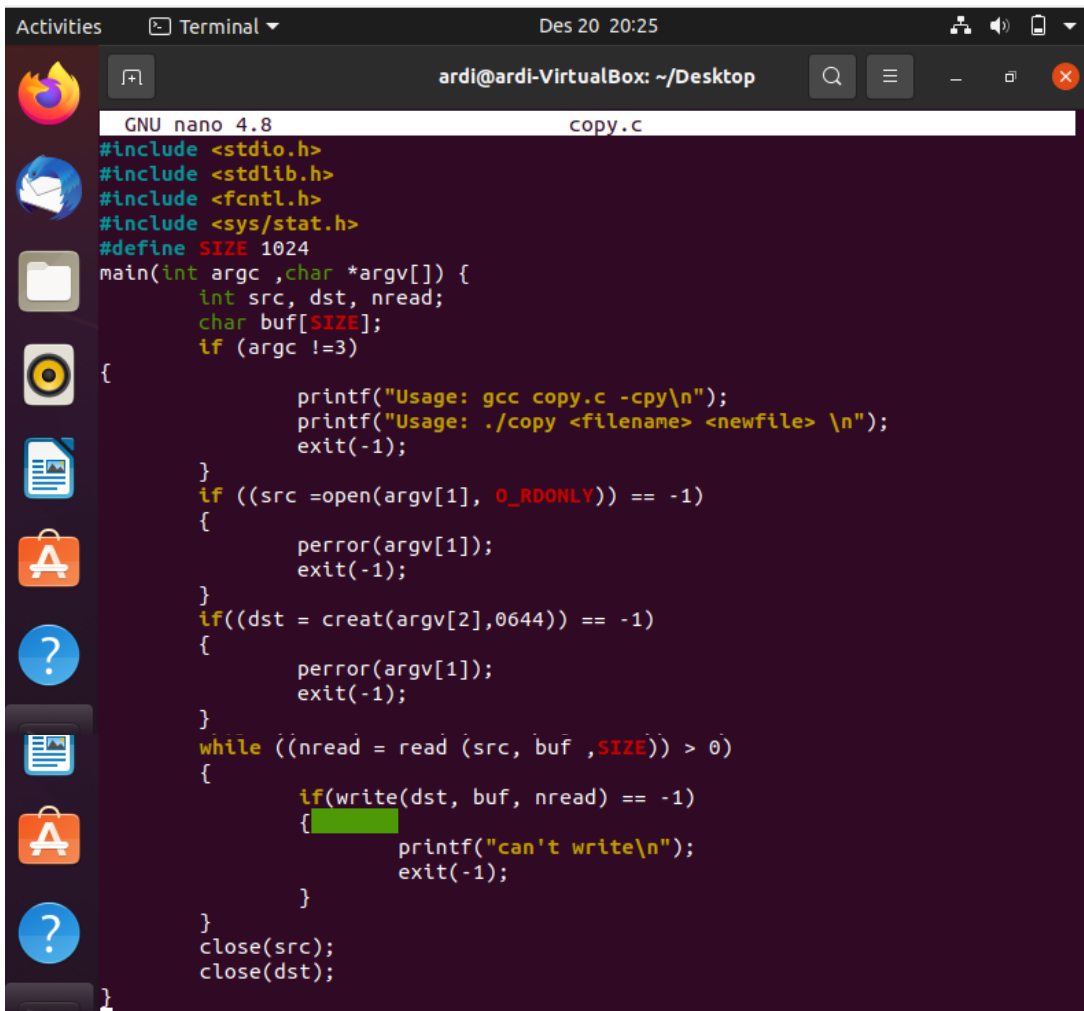
A screenshot of a terminal window titled "arddi@arddi-VirtualBox: ~/Desktop". The terminal shows the output of the command `cat file.txt`, which displays the text "arddi hergustiyan" followed by "1200210241" on the next line. The terminal window has a dark background and a light-colored text area.

Isi teks dari file.txt



A screenshot of a terminal window titled "arddi@arddi-VirtualBox: ~/Desktop". The terminal shows the following commands and their outputs: `nano mygrep.c`, `gcc -o mygrep mygrep.c` (with a warning: "warning: return type defaults to 'int' [-Wimplicit-int]"), `nano file.txt`, and `./mygrep arddi file.txt`, which outputs "arddi hergustiyan".

3. Copy.c digunakan untuk mengcopy isi dari file.txt



A screenshot of a terminal window titled "arddi@arddi-VirtualBox: ~/Desktop". The terminal shows the code for a C program named `copy.c`. The code includes headers `<stdio.h>`, `<stdlib.h>`, `<fcntl.h>`, and `<sys/stat.h>`. It defines a constant `SIZE` as 1024. The `main` function takes three arguments: `argc` and `argv`. It checks if the number of arguments is 3. If not, it prints usage information and exits. If yes, it opens the source file `argv[1]` in read-only mode. If that fails, it prints an error and exits. Then it creates a destination file `argv[2]` with permissions 0644. If that fails, it prints an error and exits. It then enters a loop where it reads data from the source file into a buffer `buf` and writes it to the destination file. If writing fails, it prints "can't write" and exits. Finally, it closes both files and exits.

Program copy.c

```
Activities Terminal Des 20 20:29
ardi@ardi-VirtualBox: ~/Desktop
ardi hergustiyan
ardi@ardi-VirtualBox:~/Desktop$ nano copy.c
ardi@ardi-VirtualBox:~/Desktop$ gcc -o copy copy.c
copy.c:6:1: warning: return type defaults to 'int' [-Wimplicit-int]
6 | main(int argc ,char *argv[]) {
  |
copy.c: In function 'main':
copy.c:25:18: warning: implicit declaration of function 'read'; did you mean 'f
read'? [-Wimplicit-function-declaration]
25 | while ((nread = read (src, buf ,SIZE)) > 0)
  |                  ^~~~~
copy.c:27:6: warning: implicit declaration of function 'write'; did you mean 'f
write'? [-Wimplicit-function-declaration]
27 | if(write(dst, buf, nread) == -1)
  |     ^~~~~
copy.c:33:2: warning: implicit declaration of function 'close'; did you mean 'p
close'? [-Wimplicit-function-declaration]
33 | close(src);
  |     ^~~~~
  fclose

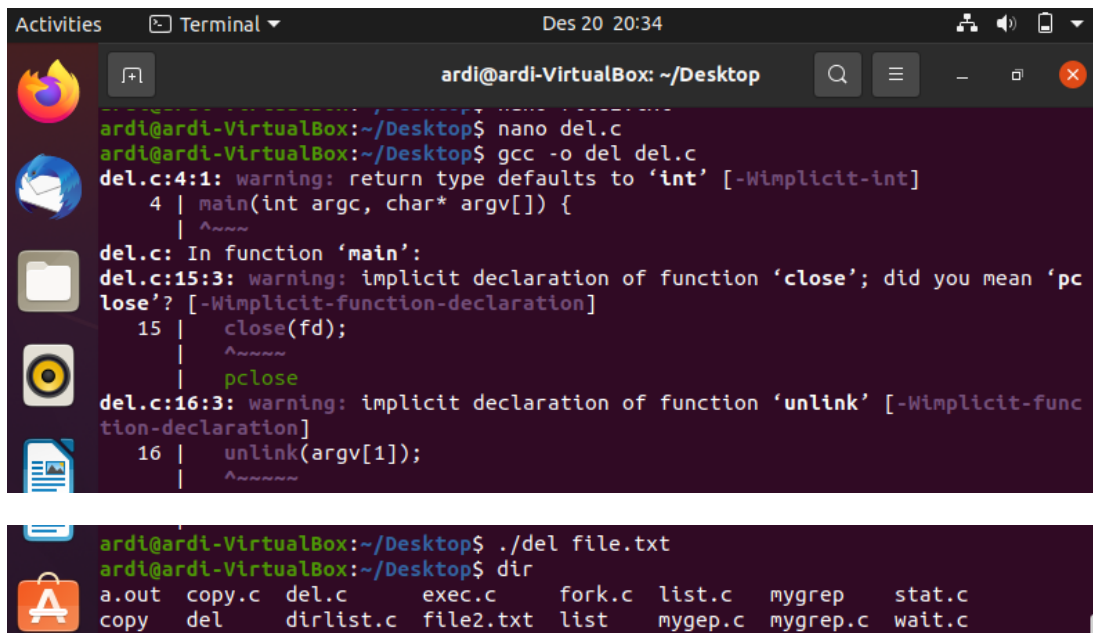
ardi@ardi-VirtualBox:~/Desktop$ ./copy file.txt file2.txt
ardi@ardi-VirtualBox:~/Desktop$ dir
a.out  del      exec.c  fork.c  mygep.c  stat.c
copy   del.c   file2.txt list  mygrep  wait.c
copy.c  dirlist.c file.txt list.c  mygrep.c
ardi@ardi-VirtualBox:~/Desktop$ nano file2.txt
ardi@ardi-VirtualBox:~/Desktop$ nano del.c
```

Proses mengCopy file.txt ke file2.txt dan melihat isi direktori apakah file2.txt sudah ada

```
Activities Terminal Des 20 20:29
ardi@ardi-VirtualBox: ~/Desktop
GNU nano 4.8 file2.txt
ardi hergustiyan
l200210241
```

Isi file2.txt sudah mencopy file.txt

4. Del.c untuk menghapus file

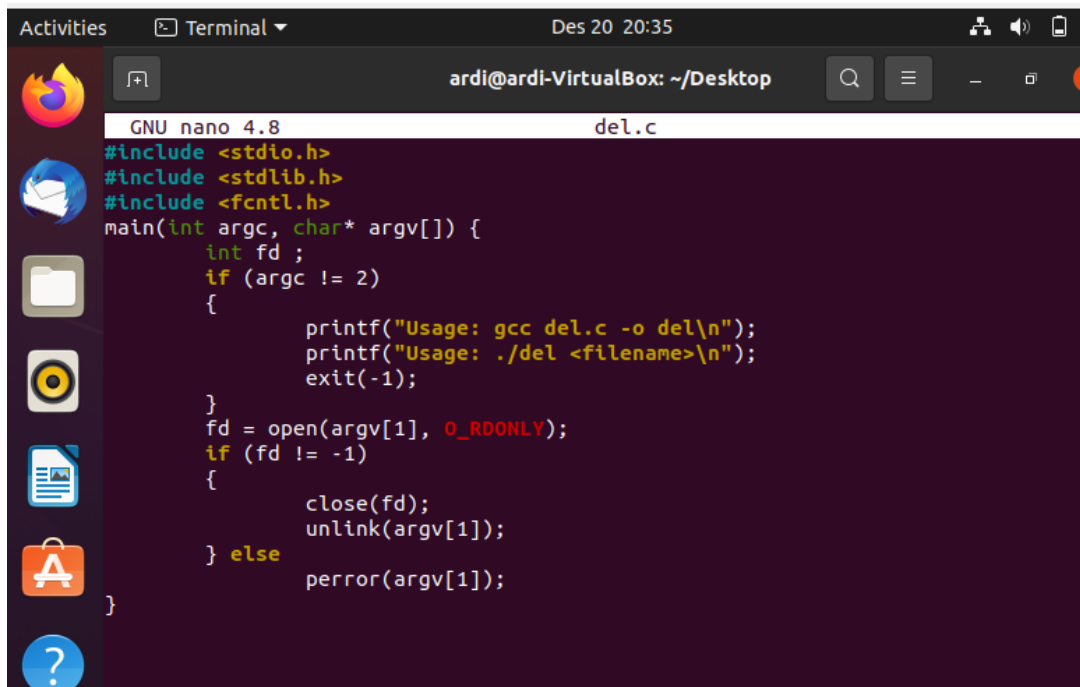


The terminal shows the user creating a file named `del.c` using `nano`. The code is a simple C program that takes a filename as an argument and attempts to delete it using `unlink`. The user then compiles the program with `gcc -o del del.c`. The compilation produces several warnings: a warning about the return type defaulting to `int`, and two warnings about implicit declarations for `close` and `unlink`. The user then runs the program with `./del file.txt`. Finally, the user runs `dir` to list the contents of the directory, showing that `file.txt` has been removed from the list.

```
ardi@ardi-VirtualBox: ~/Desktop
ardi@ardi-VirtualBox:~/Desktop$ nano del.c
ardi@ardi-VirtualBox:~/Desktop$ gcc -o del del.c
del.c:4:1: warning: return type defaults to 'int' [-Wimplicit-int]
4 | main(int argc, char* argv[]) {
  | ~~~~~
del.c: In function 'main':
del.c:15:3: warning: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-declaration]
15 |     close(fd);
    |     ~~~~~
del.c:16:3: warning: implicit declaration of function 'unlink' [-Wimplicit-function-declaration]
16 |     unlink(argv[1]);
    |     ~~~~~

ardi@ardi-VirtualBox:~/Desktop$ ./del file.txt
ardi@ardi-VirtualBox:~/Desktop$ dir
a.out  copy.c  del.c    exec.c   fork.c   list.c   mygrep   stat.c
copy   del     dirlist.c file2.txt list     mygep.c mygrep.c wait.c
```

file.txt sudah terdelete dari direktori



The terminal shows the source code of `del.c` as viewed in the `nano` editor. The code includes headers for `stdio.h`, `stdlib.h`, and `fcntl.h`. The `main` function checks the number of arguments. If there are not exactly 2 arguments, it prints a usage message and exits. If there are 2 arguments, it opens the file in read-only mode. If the opening fails, it prints an error message. If successful, it closes the file descriptor and then calls `unlink` to delete the file.

```
GNU nano 4.8                                del.c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
main(int argc, char* argv[]) {
    int fd ;
    if (argc != 2)
    {
        printf("Usage: gcc del.c -o del\n");
        printf("Usage: ./del <filename>\n");
        exit(-1);
    }
    fd = open(argv[1], O_RDONLY);
    if (fd != -1)
    {
        close(fd);
        unlink(argv[1]);
    } else
        perror(argv[1]);
}
```

Code del.c