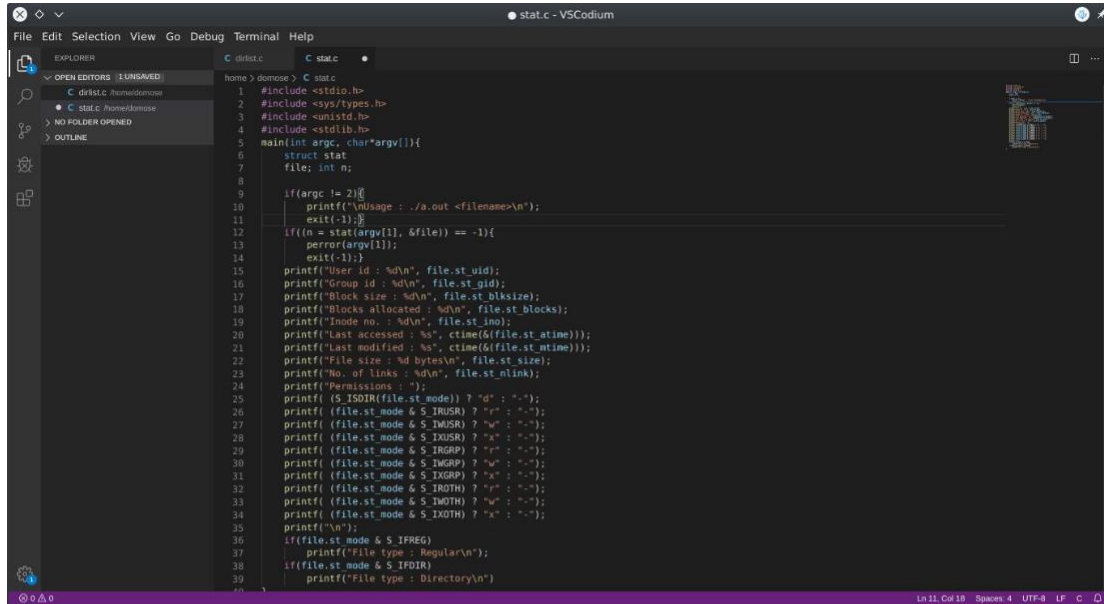


Nama : Alfian Yulianto

NIM : L200180121

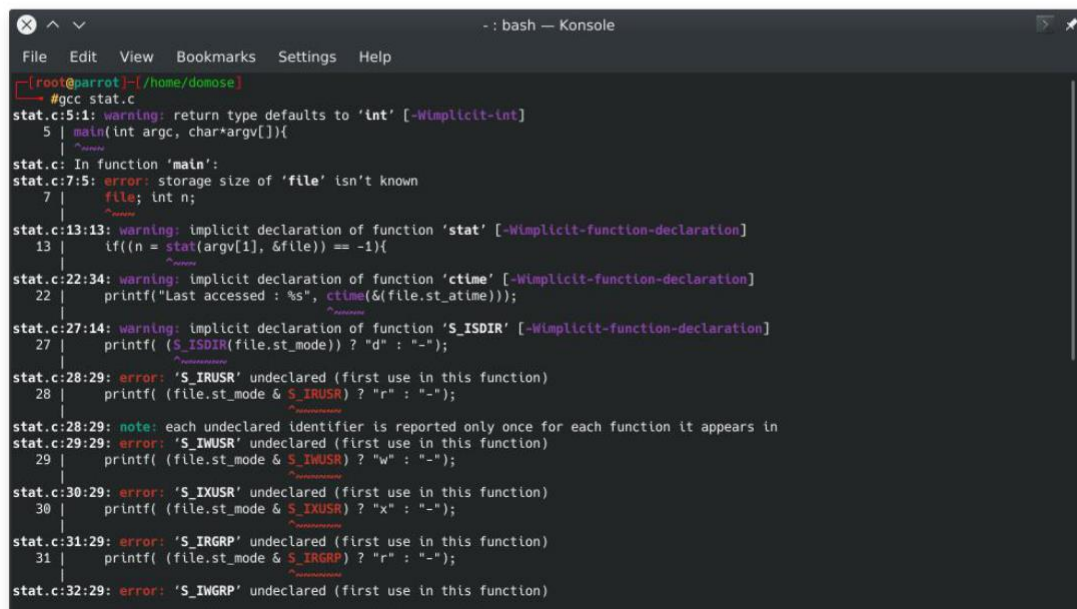
Modul 8

4. Menampilkan status file menggunakan perintah system call 'stat'.



The screenshot shows the VSCode editor with the file `stat.c` open. The code is as follows:

```
1 #include <stdio.h>
2 #include <sys/types.h>
3 #include <unistd.h>
4 #include <stdlib.h>
5 main(int argc, char*argv[]){
6     struct stat
7     file; int n;
8
9     if(argc != 2){
10         printf("\nUsage : ./a.out <filename>\n");
11         exit(-1);
12     }
13     if((n = stat(argv[1], &file)) == -1){
14         perror(argv[1]);
15         exit(-1);
16     }
17     printf("User id : %d\n", file.st_uid);
18     printf("Group id : %d\n", file.st_gid);
19     printf("Block size : %d\n", file.st_blksize);
20     printf("Blocks allocated : %d\n", file.st_blocks);
21     printf("Inode no. : %d\n", file.st_ino);
22     printf("Last accessed : %s", ctime(&file.st_atime));
23     printf("Last modified : %s", ctime(&file.st_mtime));
24     printf("File size : %d bytes\n", file.st_size);
25     printf("No. of links : %d\n", file.st_nlink);
26     printf("Permissions : ");
27     printf(" (%S_ISDIR(file.st_mode)) ? 'd' : '-');
28     printf(" (%S_ISREG(file.st_mode & S_IRUSR)) ? 'r' : '-');
29     printf(" (%S_ISCHR(file.st_mode & S_IMGR) ? 'w' : '-');
30     printf(" (%S_ISBLK(file.st_mode & S_ISBLK) ? 'b' : '-');
31     printf(" (%S_ISFIFO(file.st_mode & S_IFIFO) ? 'f' : '-');
32     printf(" (%S_ISLNK(file.st_mode & S_IFLNK) ? 'l' : '-');
33     printf(" (%S_ISOCK(file.st_mode & S_IFSOCK) ? 's' : '-');
34     printf(" (%S_ISXUGO(file.st_mode & S_IXUGO) ? 'x' : '-');
35     printf("\n");
36     if(file.st_mode & S_IFREG)
37         printf("File type : Regular\n");
38     if(file.st_mode & S_IFDIR)
39         printf("File type : Directory\n");
40 }
```



The screenshot shows a terminal window with the following output:

```
File Edit View Bookmarks Settings Help
[root@parrot:~/home/domose]
#gcc stat.c
stat.c:5:1: warning: return type defaults to 'int' [-Wimplicit-int]
5 | main(int argc, char*argv[]){
  | ^~~~~~
stat.c: In function 'main':
stat.c:7:5: error: storage size of 'file' isn't known
7 |     file; int n;
  |     ^~~~~
stat.c:13:13: warning: implicit declaration of function 'stat' [-Wimplicit-function-declaration]
13 |     if((n = stat(argv[1], &file)) == -1){
    |             ^~~~~~
stat.c:22:34: warning: implicit declaration of function 'ctime' [-Wimplicit-function-declaration]
22 |     printf("Last accessed : %s", ctime(&file.st_atime));
    |                                  ^~~~~~
stat.c:27:14: warning: implicit declaration of function 'S_ISDIR' [-Wimplicit-function-declaration]
27 |     printf(" (%S_ISDIR(file.st_mode)) ? 'd' : '-');
    |              ^~~~~~
stat.c:28:29: error: 'S_IRUSR' undeclared (first use in this function)
28 |     printf(" (%S_IRUSR(file.st_mode & S_IRUSR)) ? 'r' : '-');
    |                   ^~~~~~
stat.c:28:29: note: each undeclared identifier is reported only once for each function it appears in
stat.c:29:29: error: 'S_IWUSR' undeclared (first use in this function)
29 |     printf(" (%S_IWUSR(file.st_mode & S_IWUSR)) ? 'w' : '-');
    |                   ^~~~~~
stat.c:30:29: error: 'S_IXUSR' undeclared (first use in this function)
30 |     printf(" (%S_IXUSR(file.st_mode & S_IXUSR)) ? 'x' : '-');
    |                   ^~~~~~
stat.c:31:29: error: 'S_IRGRP' undeclared (first use in this function)
31 |     printf(" (%S_IRGRP(file.st_mode & S_IRGRP)) ? 'r' : '-');
    |                   ^~~~~~
stat.c:32:29: error: 'S_IWGRP' undeclared (first use in this function)
32 |     printf(" (%S_IWGRP(file.st_mode & S_IWGRP)) ? 'w' : '-');
    |                   ^~~~~~
```

5. Menampilkan isi direktori menggunakan perintah system call 'readdir'

The image shows a VS Code editor with a C program named `dirlist.c` open. The program is designed to list the contents of a directory. It includes `<stdio.h>`, `<dirent.h>`, and `<stdlib.h>`. The `main` function takes an array of arguments. If there are more than 2 arguments, it prints a usage message and exits. If the first argument is not a directory (checked using `opendir`), it prints an error and exits. Otherwise, it uses `readdir` to iterate through the directory entries and prints each entry's name. The program is compiled using `gcc` and executed using `./a.out`.

The terminal window shows the following commands and output:

```

[root@barrot ~]# cd /home/danose/
[root@barrot ~]# gcc dirlist.c
[root@barrot ~]# ./a.out /home/danose/
.dirrc
microsoft.gpg
Templates
Downloads
.keymap-selected
.java
.profile
.gconf
.gksu.lock
.
BurpSuite
.kde
Music
discord.deb
installid
.tdlerc
.python_history
a.out
xauthority
.emacs
.xsession-errors
gtkrc-2.0
Desktop
.vlarc
.gnupg
.pictures
.public
.wpscan
.bash_history
dirlist.c
msf4
folder_ghaib
.dbcaver4
.local
.xsession-errors.old
.
.cache
stat.c
.sqlmap
.bashrc
get-plp.py

```

