

**LAPORAN PRAKTIKUM
SISTEM OPERASI
MODUL 8
SYSTEM CALL**



Disusun Oleh :

YESY LELY YESTIANA

L200210227

Kelas E

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS KOMUNIKASI DAN INFORMATIKA
UNIVERSITAS MUHAMMADIYAH SURAKARTA
TAHUN 2022/2023**

Lembar Kerja Praktikum Modul 8

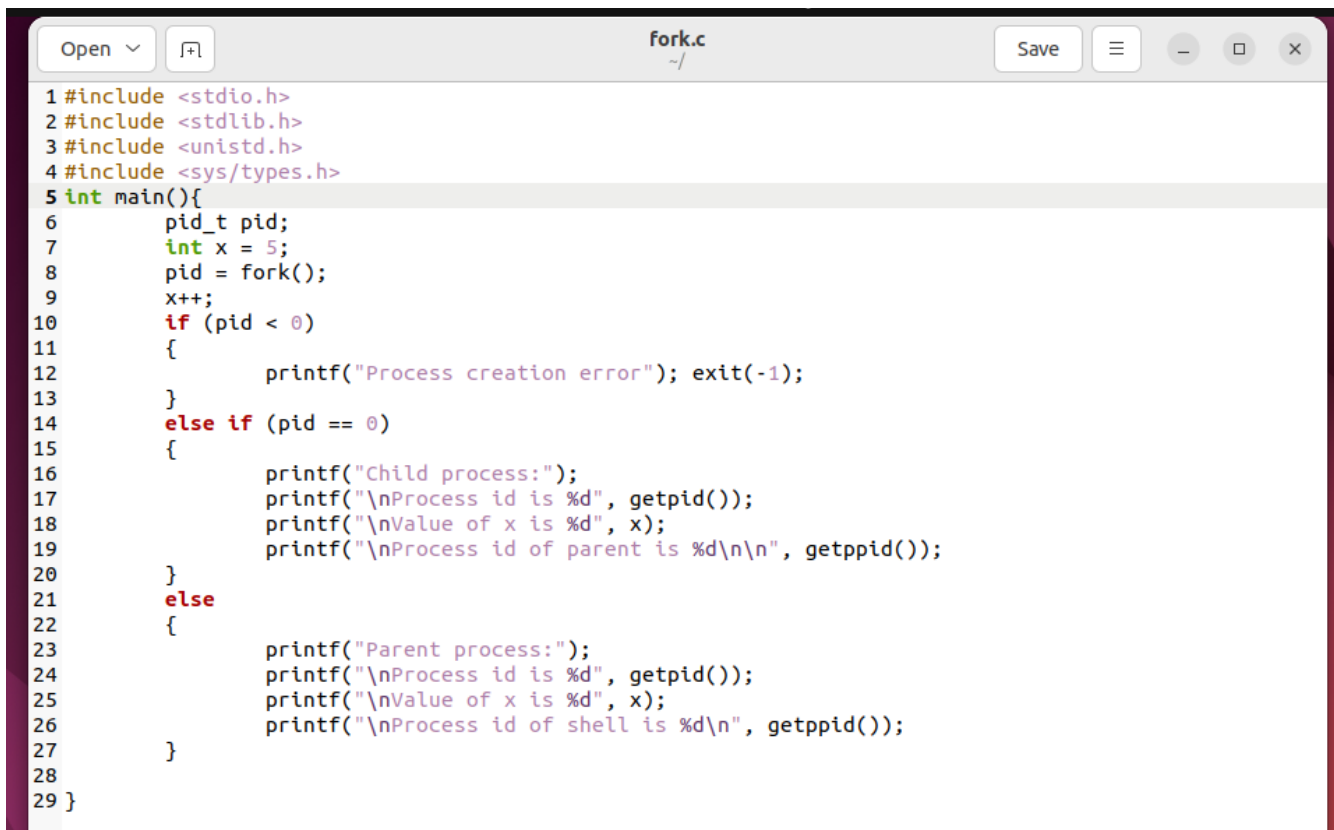
NIM	: L200210227	Nilai praktek	:
Nama	: Yesy Lely Yestiana		
Dosen Pengampu	: Heru Setiya N., ST, M.Kom	Tanda tangan	:
Nama Asisten	: -		
Tanggal Praktikum	: 07/12/2022		

Tugas!!!

Langkah Kerja:

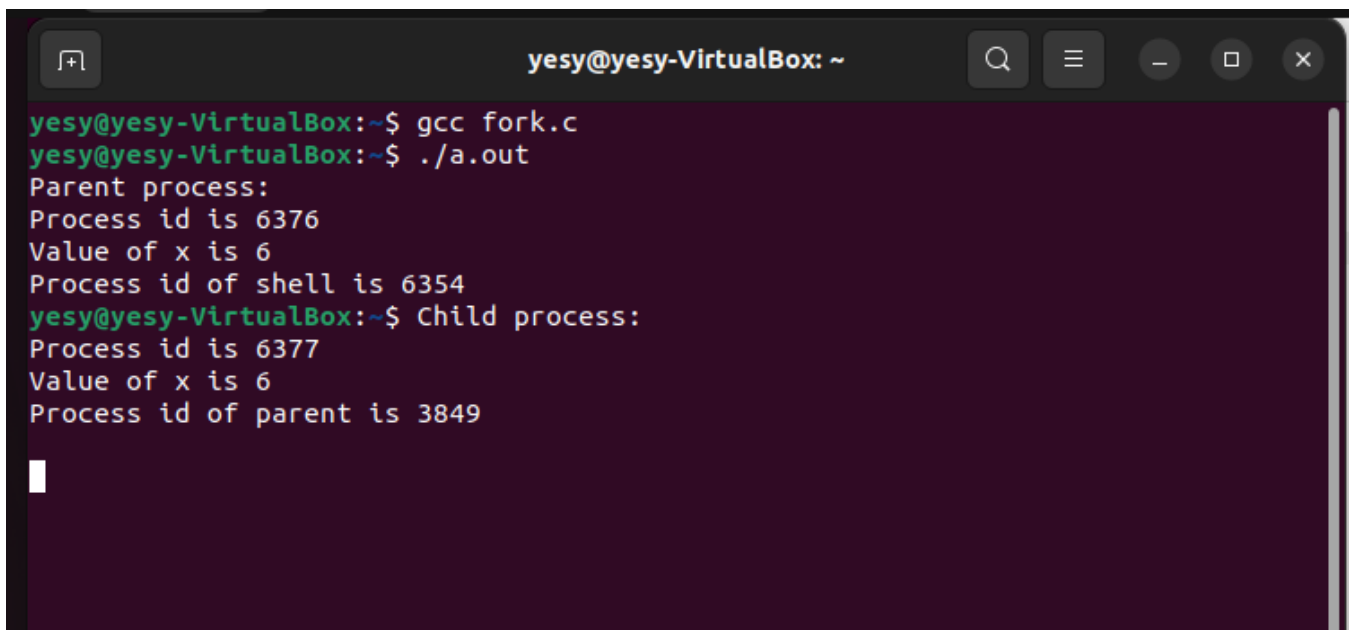
1. Membuat sebuah 'child process' (proses baru) dengan menggunakan system call 'fork'.
Membuat program dengan algoritma sebagai berikut.
(contoh program diberikan pada bagian berikutnya) :
 - a. Deklarasi sebuah variabel x yang akan diakses bersama antara child proses dan parent proses.
 - b. Membuat sebuah child proses menggunakan system call fork.
 - c. Jika return value bernilai -1, tampilkan teks 'Pembuatan proses GAGAL', dilanjutkan dengan keluar program dengan perintah system call 'exit'.
 - d. Jika return value sama dengan 0 (NOL), Tampilkan teks 'Child Process', tampilkan ID proses dari child proses menggunakan perintah system call 'getpid', tampilkan nilai x, dan tampilkan ID proses parent dengan perintah system call 'getppid'.
 - e. Untuk nilai return value yang lainnya, tampilkan teks 'Parent process', tampilkan ID dari parent proses menggunakan perintah system call getpid, tampilkan nilai x, dan tampilkan ID dari proses shell menggunakan perintah system call getpid.
 - f. Stop

- ❖ Buka terminal linux kemudian masuk ke aplikasi 'Text Editor', dengan mengetik kode program sebagai berikut :



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <sys/types.h>
5 int main(){
6     pid_t pid;
7     int x = 5;
8     pid = fork();
9     x++;
10    if (pid < 0)
11    {
12        printf("Process creation error"); exit(-1);
13    }
14    else if (pid == 0)
15    {
16        printf("Child process:");
17        printf("\nProcess id is %d", getpid());
18        printf("\nValue of x is %d", x);
19        printf("\nProcess id of parent is %d\n", getppid());
20    }
21    else
22    {
23        printf("Parent process:");
24        printf("\nProcess id is %d", getpid());
25        printf("\nValue of x is %d", x);
26        printf("\nProcess id of shell is %d\n", getppid());
27    }
28 }
29 }
```

- ❖ Hasil output an :



```
yesy@yesy-VirtualBox: ~
yesy@yesy-VirtualBox:~$ gcc fork.c
yesy@yesy-VirtualBox:~$ ./a.out
Parent process:
Process id is 6376
Value of x is 6
Process id of shell is 6354
yesy@yesy-VirtualBox:~$ Child process:
Process id is 6377
Value of x is 6
Process id of parent is 3849
```

2. Menghentikan sementara (block) proses parent sampai dengan proses child selesai, menggunakan perintah system call 'wait'.

Membuat program dengan algoritma sebagai berikut.

(Contoh program diberikan pada bagian berikutnya) :

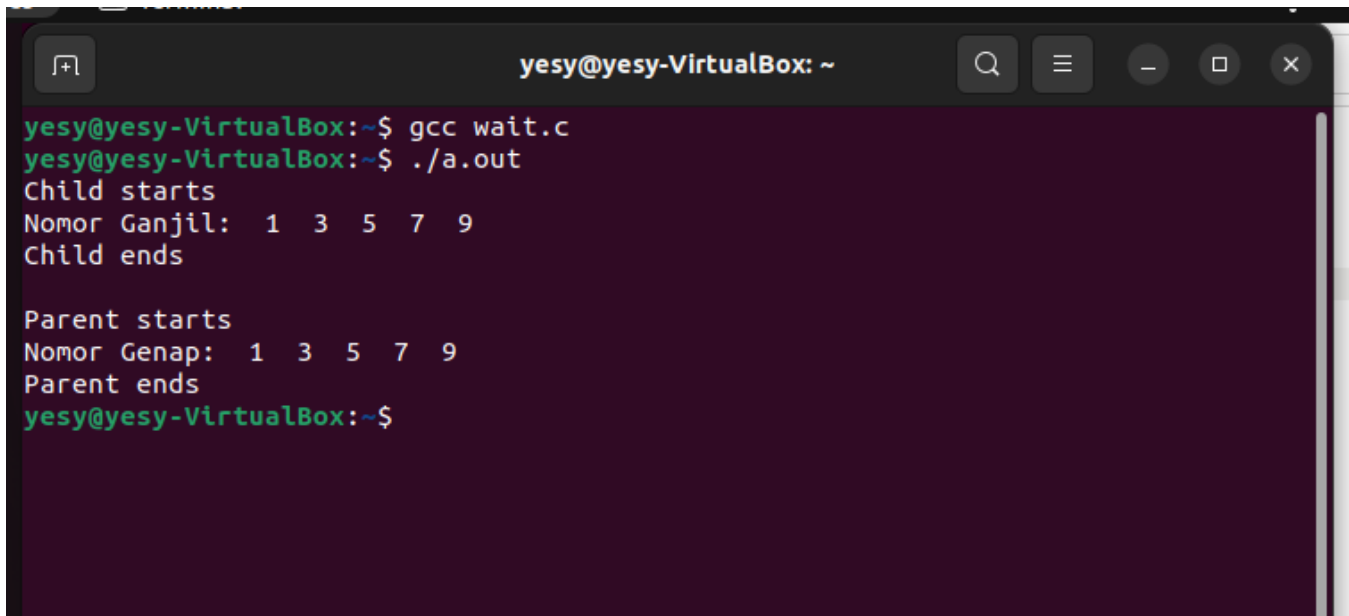
- a. Membuat sebuah child proses menggunakan system call 'fork'.
- b. Jika return value bernilai -1, selanjutnya tampilkan teks 'pembuatan proses gagal', dan keluar program dengan menggunakan perintah system call 'exit'.
- c. Jika return value berupa angka positif (> 0), 'pause' hentikan sementara 'parent' proses tunggu sampai child proses berakhir dengan menggunakan perintah system call 'wait'. Tampilkan teks 'Parent starts', selanjutnya tampilkan nomor genap mulai dari 0 s/d 10, terakhir tampilkan teks 'Parent end'.
- d. Jika return value bernilai 0 (NOL), tampilkan teks 'Child start', tampilkan nomor ganjil mulai dari 0 s/d 10, selanjutnya tampilkan teks 'child ends'.
- e. Stop.

- ❖ Buka terminal linux kemudian masuk ke aplikasi 'Text Editor', dengan mengetik kode program sebagai berikut :



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <sys/types.h>
5 #include <sys/wait.h>
6 int main() {
7     int i, status;
8     pid_t pid;
9     pid = fork();
10
11     if(pid < 0) {
12         printf("\nPembuatan proses gagal\n");
13         exit(-1);
14     }
15     else if(pid > 0)
16     {
17         wait(NULL);
18         printf ("\nParent starts\nNomor Genap:");
19         for (i=1;i<=10;i+=2)
20             printf("%3d", i);
21         printf("\nParent ends\n");
22     }
23     else if (pid == 0)
24     {
25         printf("Child starts\nNomor Ganjil:");
26         for (i=1;i<10;i+=2)
27             printf ("%3d", i);
28         printf("\nChild ends\n");
29     }
30 }
31 }
```

❖ Hasil output an :



```
yesy@yesy-VirtualBox:~$ gcc wait.c
yesy@yesy-VirtualBox:~$ ./a.out
Child starts
Nomor Ganjil: 1 3 5 7 9
Child ends

Parent starts
Nomor Genap: 1 3 5 7 9
Parent ends
yesy@yesy-VirtualBox:~$
```

3. Loading program yang dapat dieksekusi dalam sebuah 'child' proses menggunakan perintah system call 'exec'.
Membuat program dengan algoritma sebagai berikut.
(contoh program diberikan pada bagian berikutnya) :
 - a. Jika terdapat 3 argumen dalam command-line berhenti (stop).
 - b. Membuat child proses dengan perintah system call 'fork'.
 - c. Jika return value adalah -1, selanjutnya tampilkan teks 'Pembuatan proses Gagal', dan keluar program dengan perintah system call exit.
 - d. Jika return value >0 (positif), selanjutnya hentikan parent-proses sementara hingga child-proses berakhir dengan menggunakan perintah system call wait. Tampilkan teks 'Child berakhir', dan hentikan parent-proses.
 - e. Jika return value sama dengan 0 (NOL), selanjutnya tampilkan teks 'Child starts', load program dari lokasi yang diberikan dalam 'path' ke dalam child- proses, dengan menggunakan perintah system call 'exec'. Jika return value dari perintah 'exec' adalah bilangan negatif, tampilkan error yang terjadi dan stop. Hentikan child-proses.
 - f. Stop.

- ❖ Buka terminal linux kemudian masuk ke aplikasi 'Text Editor', dengan mengetik kode program sebagai berikut :

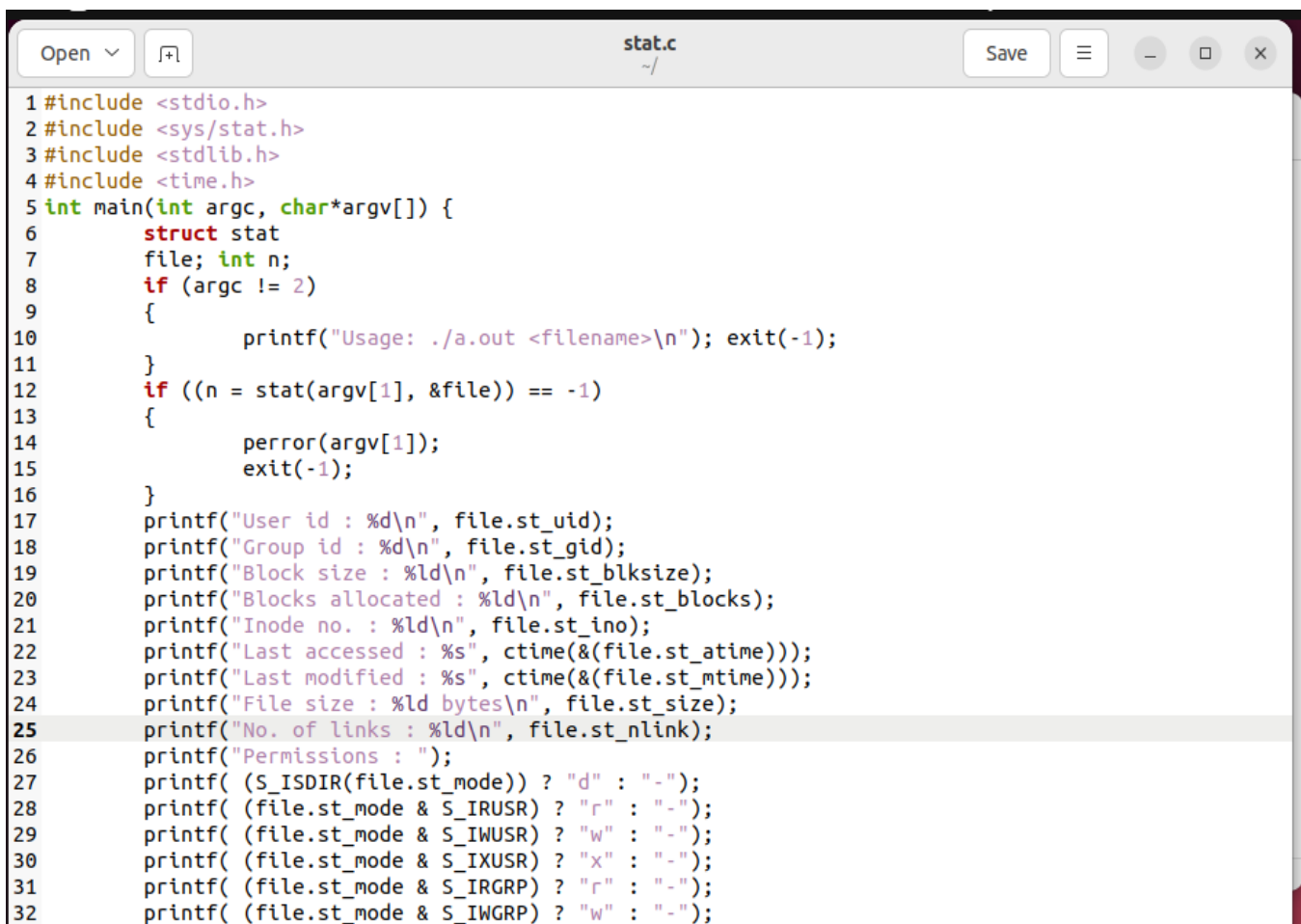
```
Open  [icon] *exec.c ~/ Save [icon] [icon] [icon] [icon]
1 #include <stdio.h>
2 #include <sys/types.h>
3 #include <unistd.h>
4 #include <stdlib.h>
5 int main(int argc, char*argv[]) {
6
7     pid_t pid;
8     int i;
9
10    if (argc !=3)
11    {
12        printf("\nInsufficient arguments to load program");
13        printf("\nUsage: ./a.out <path> <cmd>\n"); exit(-1);
14    }
15
16    switch(pid = fork())
17    {
18    case -1:
19        printf("Fork failed");
20        exit(-1);
21    case 0:
22        printf("Child process\n");
23        i = execl(argv[1], argv[2], 0);
24        if (i < 0)
25        {
26            printf("%s program not loaded using exec system call\n", argv[2]);
27            exit(-1);
28        }
29    default:
30        wait (NULL);
31        printf("Child Terminated\n");
32        exit(0);
33    }
34 }
```

- ❖ Hasil output an :

```
yesy@yesy-VirtualBox: ~$ gcc exec.c
exec.c: In function 'main':
exec.c:23:17: warning: missing sentinel in function call [-Wformat=]
   23 |             i = execl(argv[1], argv[2], 0);
       |             ^
exec.c:30:17: warning: implicit declaration of function 'wait' [-Wimplicit-function-declaration]
   30 |             wait (NULL);
       |             ^~~~
yesy@yesy-VirtualBox:~$ ./a.out /bin/ls ls
Child process
a.out      Documents  fappend.c  fork.c     Pictures   stat.c     wait.c
Desktop    Downloads  fcreate    fread.c    Public     Templates
dirlist.c  exec.c     fcreate.c  Music      snap       Videos
Child Terminated
yesy@yesy-VirtualBox:~$
```

4. Menampilkan status file menggunakan perintah system call 'stat'.
Membuat program dengan algoritma sebagai berikut.
(contoh code ada di bagian berikutnya) :
 - a. Gunakan 'nama file' yang diberikan melalui argumen dalam perintah command- line.
 - b. Jika 'nama-file' tidak ada maka stop disini (keluar program).
 - c. Panggil system call 'stat' pada 'nama-file' tersebut yang akan mengembalikan sebuah struktur.
 - d. Tampilkan informasi mengenai st_uid, st_blksize, st_block, st_size, st_nlink, etc.
 - e. Waktu dalam st_time, st_mtime dengan menggunakan fungsi ctime.
 - f. Bandingkan st_mode dengan konstanta mode seperti S_IRUSR, S_IWGRP, S_IXOTH dan tampilkan informasi mengenai 'file-permissions'.
 - g. Stop.

- ❖ Buka terminal linux kemudian masuk ke aplikasi 'Text Editor', dengan mengetik kode program sebagai berikut :



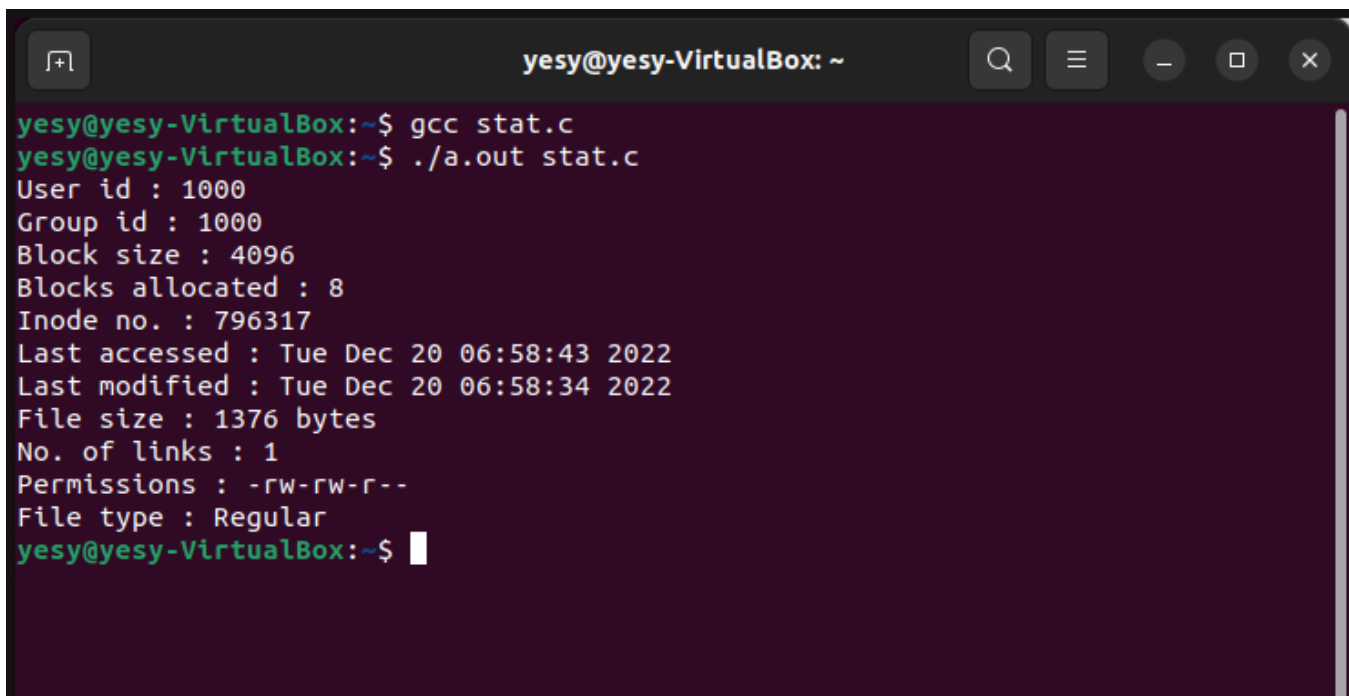
```
1 #include <stdio.h>
2 #include <sys/stat.h>
3 #include <stdlib.h>
4 #include <time.h>
5 int main(int argc, char*argv[]) {
6     struct stat
7     file; int n;
8     if (argc != 2)
9     {
10         printf("Usage: ./a.out <filename>\n"); exit(-1);
11     }
12     if ((n = stat(argv[1], &file)) == -1)
13     {
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 : %ld\n", file.st_blksize);
20     printf("Blocks allocated : %ld\n", file.st_blocks);
21     printf("Inode no. : %ld\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 : %ld bytes\n", file.st_size);
25     printf("No. of links : %ld\n", file.st_nlink);
26     printf("Permissions : ");
27     printf( (S_ISDIR(file.st_mode)) ? "d" : "-");
28     printf( (file.st_mode & S_IRUSR) ? "r" : "-");
29     printf( (file.st_mode & S_IWUSR) ? "w" : "-");
30     printf( (file.st_mode & S_IXUSR) ? "x" : "-");
31     printf( (file.st_mode & S_IRGRP) ? "r" : "-");
32     printf( (file.st_mode & S_IWGRP) ? "w" : "-");
```

```

25     printf("No. of links : %ld\n", file.st_nlink);
26     printf("Permissions : ");
27     printf( (S_ISDIR(file.st_mode)) ? "d" : "-");
28     printf( (file.st_mode & S_IRUSR) ? "r" : "-");
29     printf( (file.st_mode & S_IWUSR) ? "w" : "-");
30     printf( (file.st_mode & S_IXUSR) ? "x" : "-");
31     printf( (file.st_mode & S_IRGRP) ? "r" : "-");
32     printf( (file.st_mode & S_IWGRP) ? "w" : "-");
33     printf( (file.st_mode & S_IXGRP) ? "x" : "-");
34     printf( (file.st_mode & S_IROTH) ? "r" : "-");
35     printf( (file.st_mode & S_IWOTH) ? "w" : "-");
36     printf( (file.st_mode & S_IXOTH) ? "x" : "-");
37     printf("\n");
38     if(file.st_mode & S_IFREG)
39         printf("File type : Regular\n");
40     if(file.st_mode & S_IFDIR)
41         printf("File type : Directory\n");
42 }
43

```

❖ Hasil output an :

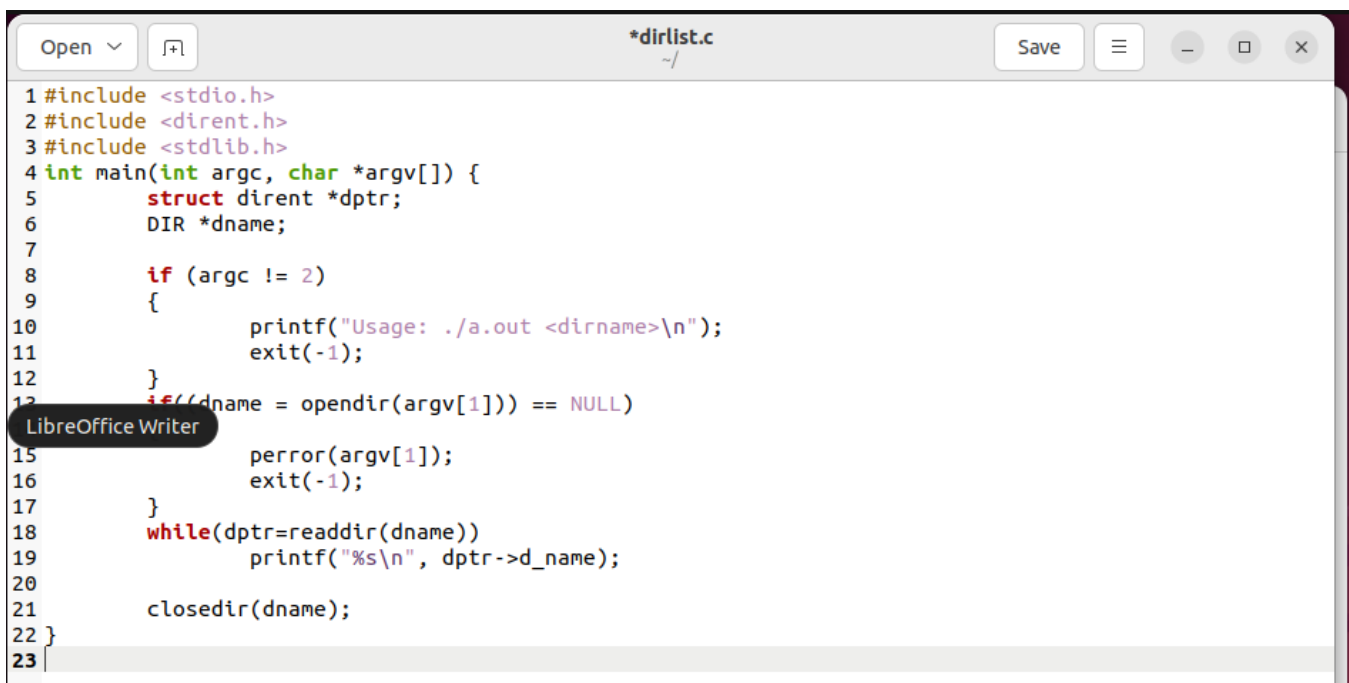


```

yesy@yesy-VirtualBox: ~
yesy@yesy-VirtualBox:~$ gcc stat.c
yesy@yesy-VirtualBox:~$ ./a.out stat.c
User id : 1000
Group id : 1000
Block size : 4096
Blocks allocated : 8
Inode no. : 796317
Last accessed : Tue Dec 20 06:58:43 2022
Last modified : Tue Dec 20 06:58:34 2022
File size : 1376 bytes
No. of links : 1
Permissions : -rw-rw-r--
File type : Regular
yesy@yesy-VirtualBox:~$

```


5. Menampilkan isi direktori menggunakan perintah system call 'readdir'
- Membuat program dengan algoritma sebagai berikut.
- (contoh code ada di bagian berikutnya) :
- Gunakan 'nama-direktori' yang diberikan sebagai argumen pada command-line.
 - Jika direktori tidak ditemukan stop, keluar program.
 - Buka direktori menggunakan perintah system call 'opendir' yang akan menghasilkan sebuah struktur.
 - Baca direktori menggunakan perintah system call 'readdir' yang juga akan menghasilkan struktur data.
 - Tampilkan d_name (nama direktori).
 - Akhiri pembacaan direktori dengan perintah system call 'closedir'.
 - Stop.
- ❖ Buka terminal linux kemudian masuk ke aplikasi 'Text Editor', dengan mengetik kode program sebagai berikut :



The screenshot shows a text editor window titled '*dirlist.c' with a file icon and 'Save' button. The code is as follows:

```
1 #include <stdio.h>
2 #include <dirent.h>
3 #include <stdlib.h>
4 int main(int argc, char *argv[]) {
5     struct dirent *dptr;
6     DIR *dname;
7
8     if (argc != 2)
9     {
10         printf("Usage: ./a.out <dirname>\n");
11         exit(-1);
12     }
13     if ((dname = opendir(argv[1])) == NULL)
14     {
15         perror(argv[1]);
16         exit(-1);
17     }
18     while (dptr = readdir(dname))
19         printf("%s\n", dptr->d_name);
20
21     closedir(dname);
22 }
23
```

A 'LibreOffice Writer' tooltip is visible over line 13.

❖ Hasil output an :

```
yesy@yesy-VirtualBox: ~  
yesy@yesy-VirtualBox:~$ gcc dirlist.c  
yesy@yesy-VirtualBox:~$ ./a.out /home/yesy  
fork.c  
.config  
dirlist.c  
.ssh  
.cache  
.gnupg  
.local  
.sudo_as_admin_successful  
..  
snap  
fread.c  
fcreate.c  
Desktop  
exec.c  
.bash_logout  
fcreate  
a.out  
wait.c  
Public  
Downloads  
Music  
.bashrc  
.bash_history  
fappend.c  
stat.c  
.profile  
Pictures  
Documents  
.  
Videos  
.audacity-data  
Templates  
yesy@yesy-VirtualBox:~$
```