# LAPORAN PRAKTIKUM
# SISTEM OPERASI


Disusun oleh :

IRVIANTI DWITYARA SANY

L200210251

**PROGRAM STUDI TEKNIK INFROMATIKA**

**FAKULTAS KOMUNIKASI DAN INFORMATIKA**

**UNIVERSITAS MUHAMMADIYAH SURAKARTA**

**TAHUN 2022/2023**

**Lembar Kerja Modul 8**

| | | |
|---|---|---|
| Nama | : Irvianti Dwityara Sany | Nilai Praktek : |
| NIM | : L200210251 | |
| Nama Asisten | : | Tanda Tangan : |
| Tanggal Praktikum | : Selasa, 6 Desember 2022 | |

1. Membuat sebuah Child Process dengan menggunakan system call “fork”

**Kode Program :**

```
ipik@ipik-VirtualBox:~$ vi fork.c
ipik@ipik-VirtualBox:~$ gcc fork.c
```

Terminal — Des 6 08:46 — ipik@ipik-VirtualBox: ~

```
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
main() {
        pid_t pid;
        int x = 5;
        pid = fork();
        x++;
        if (pid < 0){
                printf("Proses creation error");
                exit(-1);
        }
        else if (pid == 0){
                printf("Child process");
                printf("\nProcess id is %d", getpid());
                printf("\nValue of x is %d, x");
                printf("Process id of parent is %d\n\n", getppid());
        }
        else{
                printf("\nParent process");
                printf("\nProcess id is %d", getppid());
                printf("\nValue of x is %d", x);
                printf("\nProcess id of shell is %d\n", getppid)
        }
}
~
~
~
-- INSERT --                                              26,2          Bot
```

```
ipik@ipik-VirtualBox:~$ ./a.out

Parent process
Process id is 2466
Value of x is 6
Process id of shell is -1528794944
ipik@ipik-VirtualBox:~$ Child process
Process id is 5093
Value of x is -1960587207, xProcess id of parent is 1373
```

2. Menghentikan sementara (block) proses parent sampai dengan proses chile selesai, menggunakan perintah system call “wait”

**Kode Program :**

```
ipik@ipik-VirtualBox:~$ vi wait.c
ipik@ipik-VirtualBox:~$ gcc exec.c
```

Terminal Des 6 09:23

ipik@ipik-VirtualBox: ~

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
main(){
        int i, status;
        pid_t pid;
        pid = fork();

        if (pid < 0){
                printf("\nPembuatan proses gagal\n");
                exit(-1);
        }
        else if (pid > 0)
        {
                wait(NULL);
                printf("\nParent starts\nNomor Genap :");
                for (i=2; i<=10; i+=2)
                        printf("%3d", i);
                printf("\nParent ends\n");
        }
        else if (pid == 0)
        {
                printf ("Child starts\nNomor Ganjil :");
                for (i=1; i<10; i+=2)
                        printf("%3d", i);
        }
-- INSERT --                                                    5,10
```

```
ipik@ipik-VirtualBox:~$ ./a.out
Child starts
Nomor Ganjil :  1  3  5  7  9
Parent starts
Nomor Genap :  2  4  6  8 10
Parent ends
ipik@ipik-VirtualBox:~$
```

3. Loading program yang dapat dieksekusi dalam sebuah ‘child’ proses menggunkan perintah system call ‘exec’

**Kode Program :**

```
ipik@ipik-VirtualBox:~$ vi exec.c
ipik@ipik-VirtualBox:~$ gcc exec.c
```

Terminal  Des 6 09:37

Files  ipik@ipik-VirtualBox: ~

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>
main(int argc, char*argv[]){
        pid_t pid;
        int i;

        if (argc != 3)
        {
                printf("\nInsufficient arguments to load program");
                printf("\nUsage: ./a.out <path> <cmd>\n"); exit(-1);
        }

        switch (pid = fork())
        {
        case -1:
                printf("Fork failed");
                exit(-1);
        case 0:
                printf("Child process\n");
                i = execl(argv[1], argv[2], 0);
                if (i < 0)
                {
                printf("%s program not loaded using exec system call\n", argv[2
]);
                exit (-1);
                }
```

```
                }
        default:
                wait(NULL);
                printf("Child Terminated\n");
                exit(0);
        }
}
:wq
```

```
ipik@ipik-VirtualBox:~$ ./a.out /bin/ls ls
Child process
a.out    Documents  exec.c  Music     Public  Templates  wait.c
Desktop  Downloads  fork.c  Pictures  snap    Videos
Child Terminated
ipik@ipik-VirtualBox:~$
```

4. Menampilkan status file menggunakan perintah system call ‘stat’

**Kode Program :**

```
ipik@ipik-VirtualBox:~$ vi stat.c
ipik@ipik-VirtualBox:~$ gcc stat.c
```

Terminal  Des 6 10:44

ipik@ipik-VirtualBox: ~

```
#include <stdio.h>
#include <sys/stat.h>
#include<stdlib.h>
#include <time.h>
int main(int argc, char*argv[]){
        struct stat
        file; int n;
        if (argc != 2)
        {
                printf("Usage: ./a.out <filename>\n"); exit(-1);
        }
        if ((n = stat(argv[1], &file)) == -1)
        {
                perror(argv[1]);
                exit(-1);
        }
        printf("User id : %d\n", file.st_uid);
        printf("Group id : %d\n", file.st_gid);
        printf("Block size : %d\n", file.st_blksize);
        printf("Blocks allocated : %d\n", file.st_blocks);
        printf("Inode no. : %s\n", file.st_ino);
        printf("Last accesed : %s", ctime(&(file.st_atime)));
        printf("Last modified : %s", ctime(&(file.st_mtime)));
        printf("File size : %d bytes\n", file.st_size);
        printf("No. of links : %d\n", file.st_nlink);
        printf("Permissions :");
        printf( (S_ISDIR(file.st_mode)) ? "d" : "-");
        printf( (file.st_mode & S_IRUSR) ? "r" : "-");
```

```
        printf( (file.st_mode & S_IWUSR) ? "w" : "-");
        printf( (file.st_mode & S_IXUSR) ? "x" : "-");
        printf( (file.st_mode & S_IRGRP) ? "r" : "-");
        printf( (file.st_mode & S_IWUSR) ? "w" : "-");
        printf( (file.st_mode & S_IXUSR) ? "x" : "-");
        printf( (file.st_mode & S_IROTH) ? "r" : "-");
        printf( (file.st_mode & S_IWOTH) ? "w" : "-");
        printf( (file.st_mode & S_IXOTH) ? "x" : "-");
        printf("\n");
        if (file.st_mode & S_IFREG)
                printf("File type : Regular\n");
        if (file.st_mode & S_IFDIR)
                printf ("File type : Directory\n");
}
:wq
```

```
ipik@ipik-VirtualBox:~$ ./a.out stat.c
User id : 1000
Group id : 1000
Block size : 4096
Blocks allocated : 8
Segmentation fault (core dumped)
ipik@ipik-VirtualBox:~$
```

5. Menampilkan isi direktori menggunakan perintah system call ‘readdir’

**Kode Program :**

```
ipik@ipik-VirtualBox:~$ vi dirlist.c
ipik@ipik-VirtualBox:~$ gcc dirlist.c
```

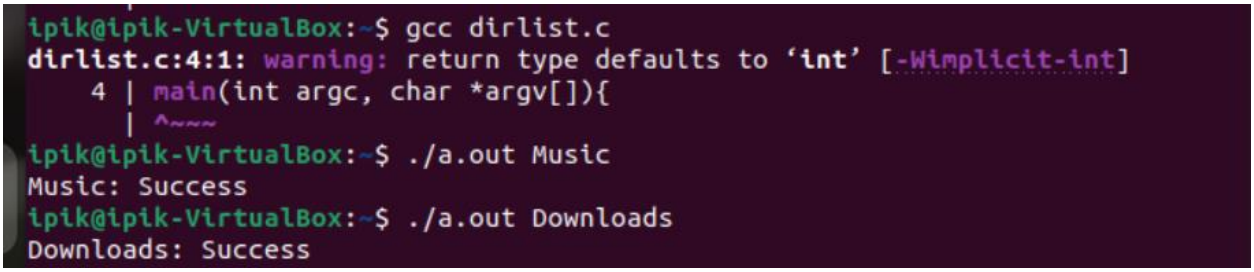
Terminal Des 6 10:56

ipik@ipik-VirtualBox: ~

```
#include <stdio.h>
#include <dirent.h>
#include <stdlib.h>
main(int argc, char *argv[]){
        struct dirent *dptr;
        DIR *dname;

        if (argc != 2)
        {
                printf ("Usage: ./a.out <dirname>\n");
                exit(-1);
        }
        if ((dname = opendir(argv[1] == NULL)));
        {
                perror(argv[1]);
                exit(-1);
        }
        while(dpt=readdir(dname))
                printf ("%s\n", dptr->d_name);

        closedir(dname);
}
```


ipik@ipik-VirtualBox:~$ gcc dirlist.c
dirlist.c:4:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
4 | main(int argc, char *argv[]){
| ^~~~
ipik@ipik-VirtualBox:~$ ./a.out Music
Music: Success
ipik@ipik-VirtualBox:~$ ./a.out Downloads
Downloads: Success