Ilyas Raihan Nadhif
L200180119
C

exec.c


```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
int 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]);
```



```c
    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);
    }
}
```



```
root@kali:~/modul8# gcc exec.c
exec.c: In function 'main':
exec.c:31:9: warning: implicit declaration of function 'wait'; did you mean 'main
'? [-Wimplicit-function-declaration]
        wait(NULL);
        ^~~~
        main
root@kali:~/modul8# ./a.out /bin/ls ls
Child process
a.out    contoh.txt  exec     fork.c   stat.c
contoh   dirlist.c   exec.c   stat     wait.c
Child Terminated
root@kali:~/modul8#
```

stat.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <sys/stat.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("Block allocated: %d\n", file.st_blocks);
    printf("Inode no. : %d\n", file.st_ino);
    printf("Last accessed : %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("Permission : ");
```

```c
    printf("Last accessed : %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("Permission : ");
    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_IWGRP) ? "w" : "-");
    printf( (file.st_mode & S_IXGRP) ? "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 : Reguler\n");
    }
    if (file.st_mode & S_IFDIR)
    {
        printf("File type: Directory\n");
    }
}
```

```
root@kali:~/Dokumen# gcc stat.c
root@kali:~/Dokumen# ./a.out stat.c
User id : 0
Group id: 0
Block size: 4096
Block allocated: 8
Inode no. : 961
Last accessed : Mon Dec  2 23:00:00 2019
Last modified : Tue Nov 19 13:38:19 2019
File size : 1519 bytes
No. of links : 1
Permission : -rw-r--r--
File type : Reguler
root@kali:~/Dokumen#
```

dirlist.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <dirent.h>
int 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 (dptr=readdir(dname))
    {
        printf("%s\n", dptr->d_name);
    }
    closedir(dname);
}
```

```
root@kali:~/modul8# gcc dirlist.c
root@kali:~/modul8# ./a.out /
initrd.img
mnt
vmlinuz.old
sbin
usr
proc
var
tmp
boot
.cache
initrd.img.old
lib
root
a.out
lost+found
dev
bin
Modul8
home
lib32
srv
vmlinuz
sys
lib64
media
libx32
run
```

```
usr
proc
var
tmp
boot
.cache
initrd.img.old
lib
root
a.out
lost+found
dev
bin
Modul8
home
lib32
srv
vmlinuz
sys
lib64
media
libx32
run
etc
..
opt
.
root@kali:~/modul8#
```