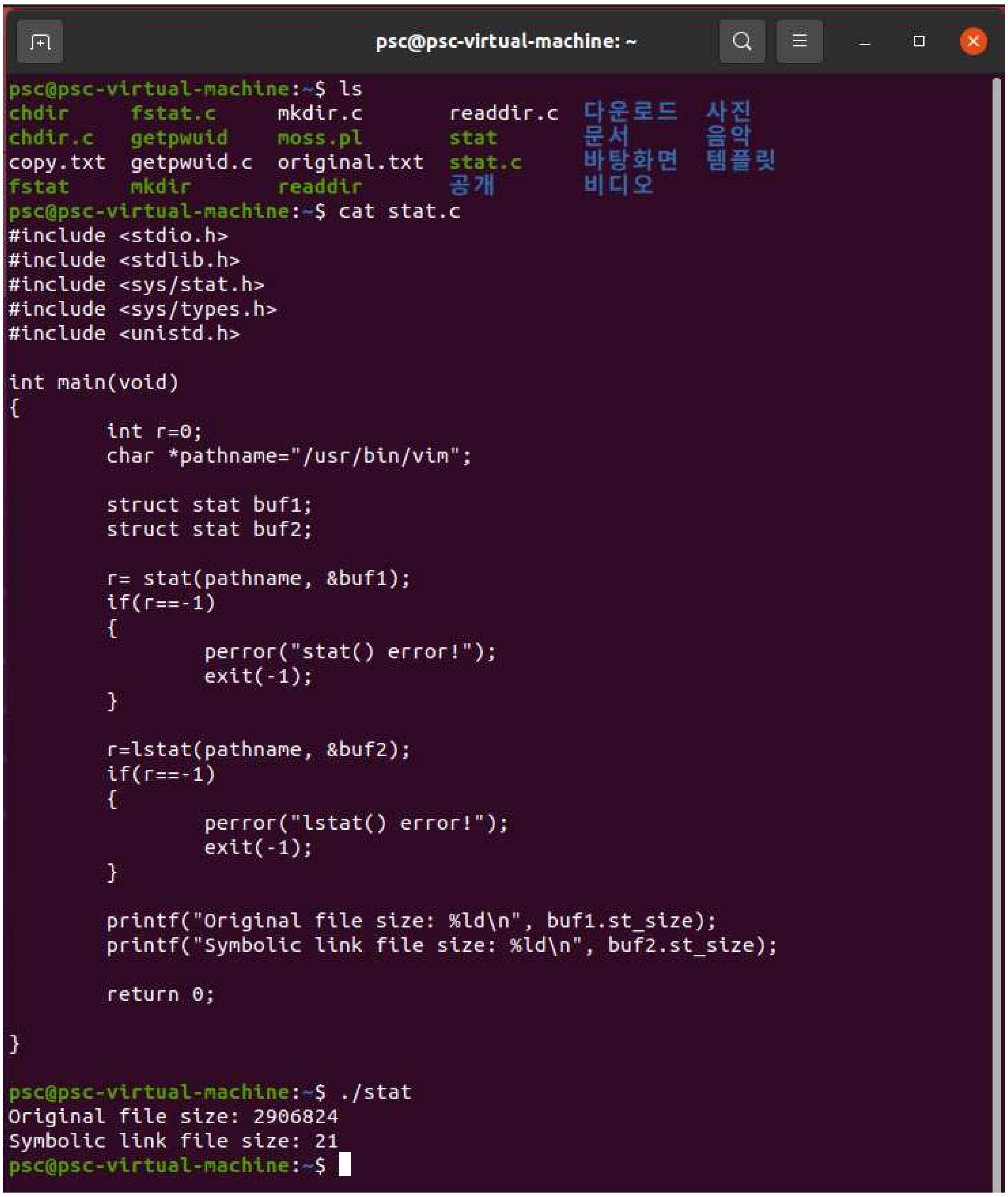
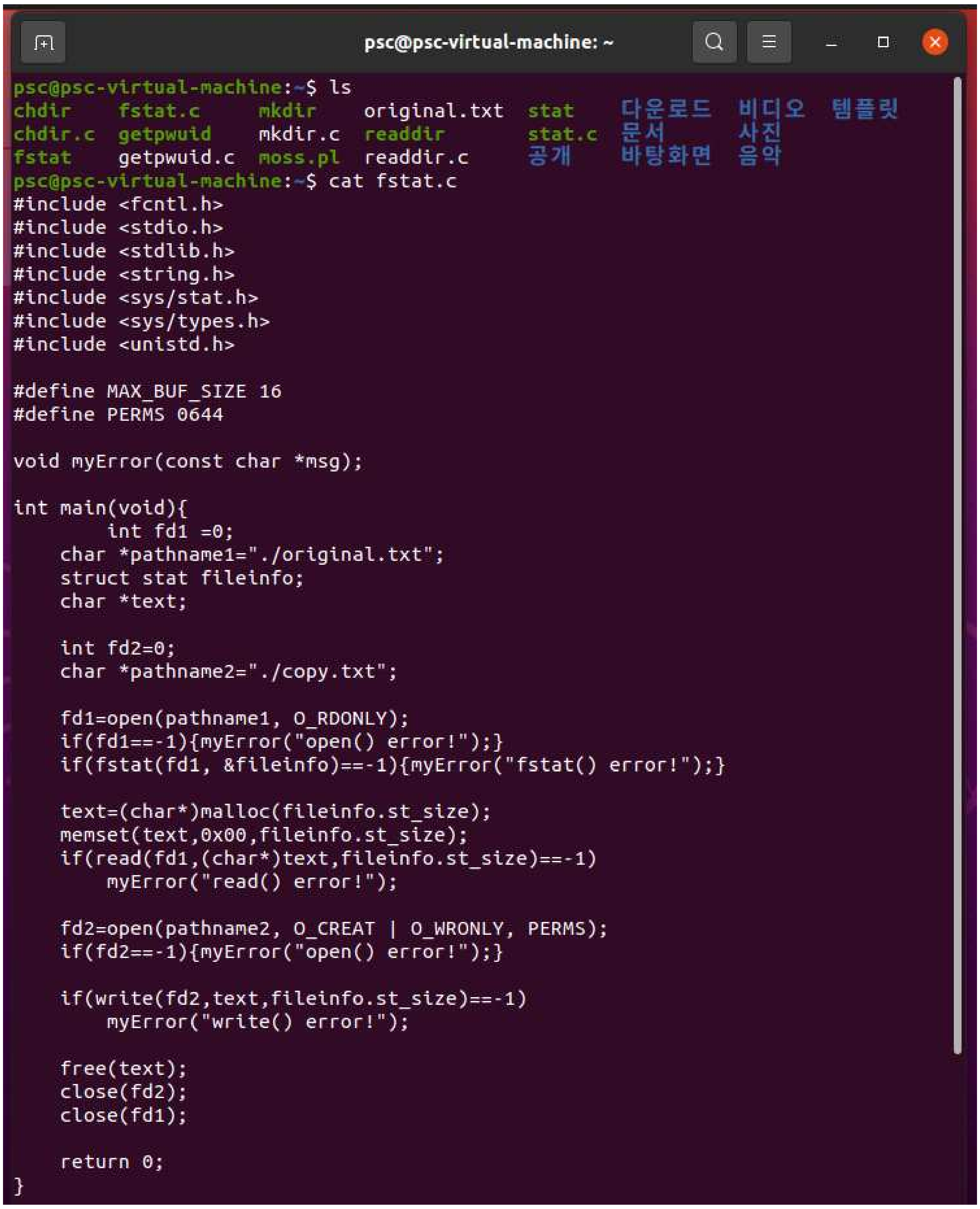
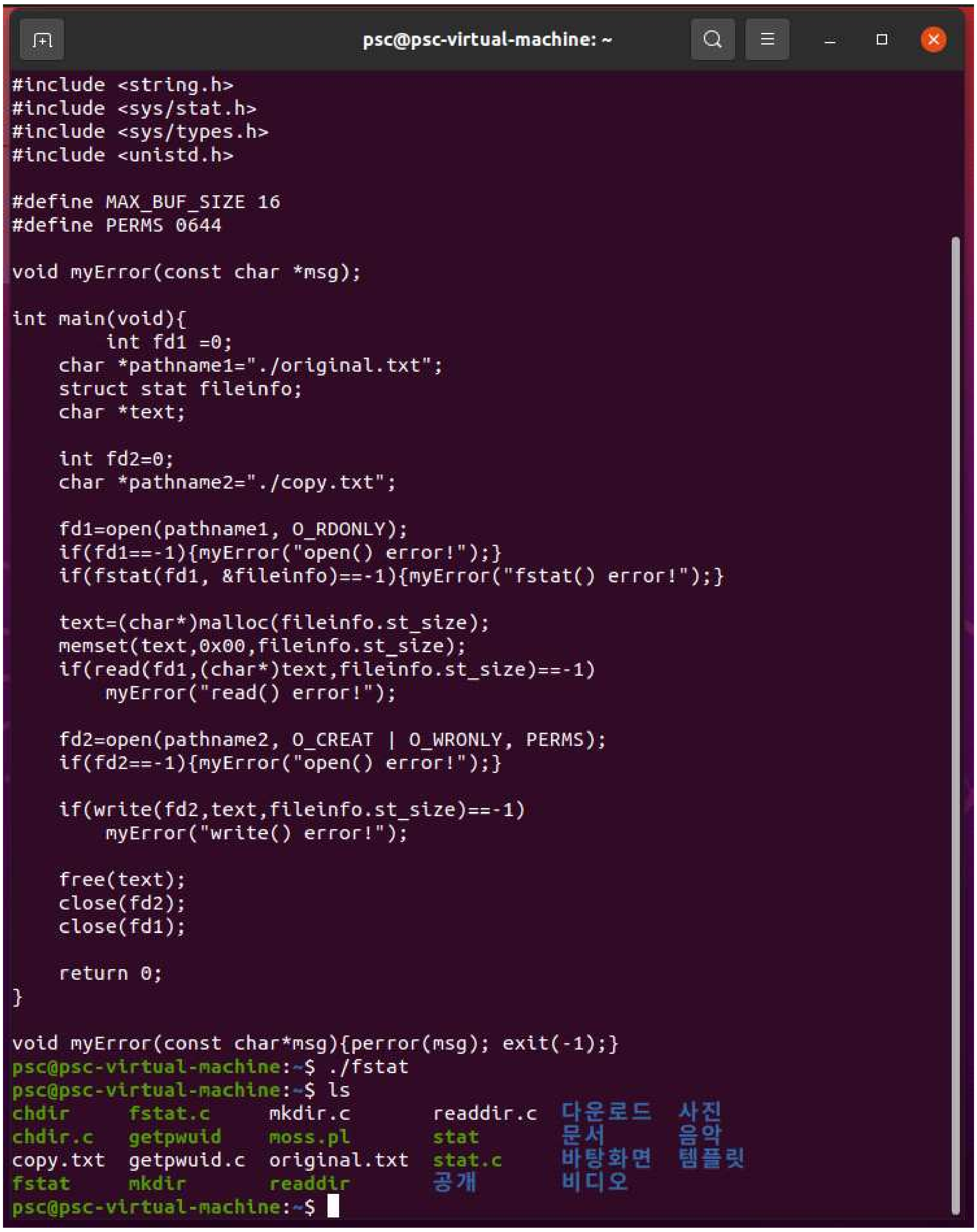
2020203002 박상천



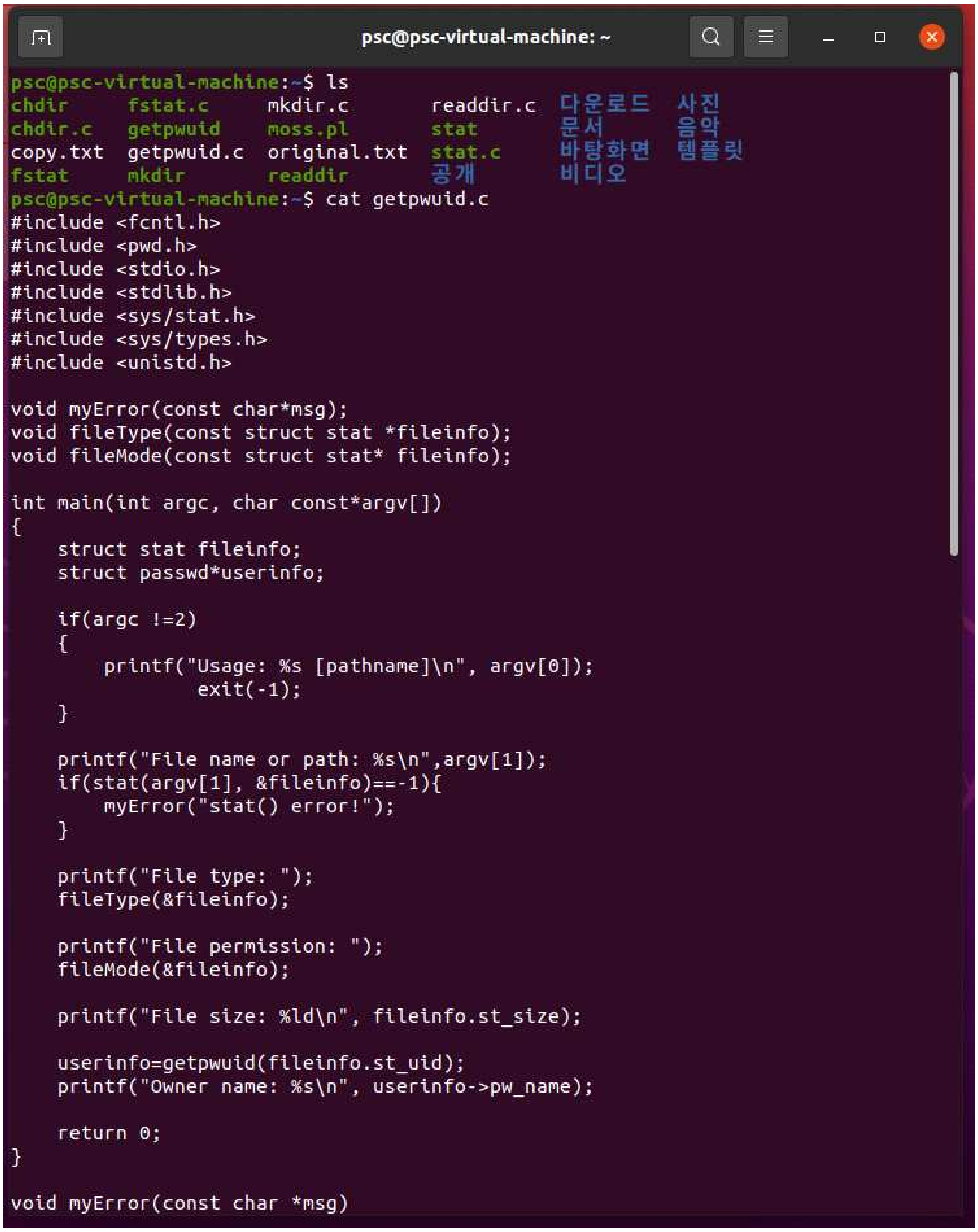
```
psc@psc-virtual-machine:~$ ls
chdir      fstat.c     mkdir.c       readdir.c  다운로드  사진
chdir.c    getpwuid    moss.pl       stat       문서      음악
copy.txt   getpwuid.c  original.txt  stat.c     바탕화면  템플릿
fstat      mkdir       readdir       공개       비디오
psc@psc-virtual-machine:~$ cat stat.c
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

int main(void)
{
        int r=0;
        char *pathname="/usr/bin/vim";

        struct stat buf1;
        struct stat buf2;

        r= stat(pathname, &buf1);
        if(r==-1)
        {
                perror("stat() error!");
                exit(-1);
        }

        r=lstat(pathname, &buf2);
        if(r==-1)
        {
                perror("lstat() error!");
                exit(-1);
        }

        printf("Original file size: %ld\n", buf1.st_size);
        printf("Symbolic link file size: %ld\n", buf2.st_size);

        return 0;

}

psc@psc-virtual-machine:~$ ./stat
Original file size: 2906824
Symbolic link file size: 21
psc@psc-virtual-machine:~$
```



```
psc@psc-virtual-machine:~$ ls
chdir     fstat.c     mkdir     original.txt  stat     다운로드  비디오  템플릿
chdir.c   getpwuid    mkdir.c   readdir       stat.c   문서      사진
fstat     getpwuid.c  moss.pl   readdir.c     공개     바탕화면  음악
psc@psc-virtual-machine:~$ cat fstat.c
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

#define MAX_BUF_SIZE 16
#define PERMS 0644

void myError(const char *msg);

int main(void){
        int fd1 =0;
    char *pathname1="./original.txt";
    struct stat fileinfo;
    char *text;

    int fd2=0;
    char *pathname2="./copy.txt";

    fd1=open(pathname1, O_RDONLY);
    if(fd1==-1){myError("open() error!");}
    if(fstat(fd1, &fileinfo)==-1){myError("fstat() error!");}

    text=(char*)malloc(fileinfo.st_size);
    memset(text,0x00,fileinfo.st_size);
    if(read(fd1,(char*)text,fileinfo.st_size)==-1)
        myError("read() error!");

    fd2=open(pathname2, O_CREAT | O_WRONLY, PERMS);
    if(fd2==-1){myError("open() error!");}

    if(write(fd2,text,fileinfo.st_size)==-1)
        myError("write() error!");

    free(text);
    close(fd2);
    close(fd1);

    return 0;
}
```

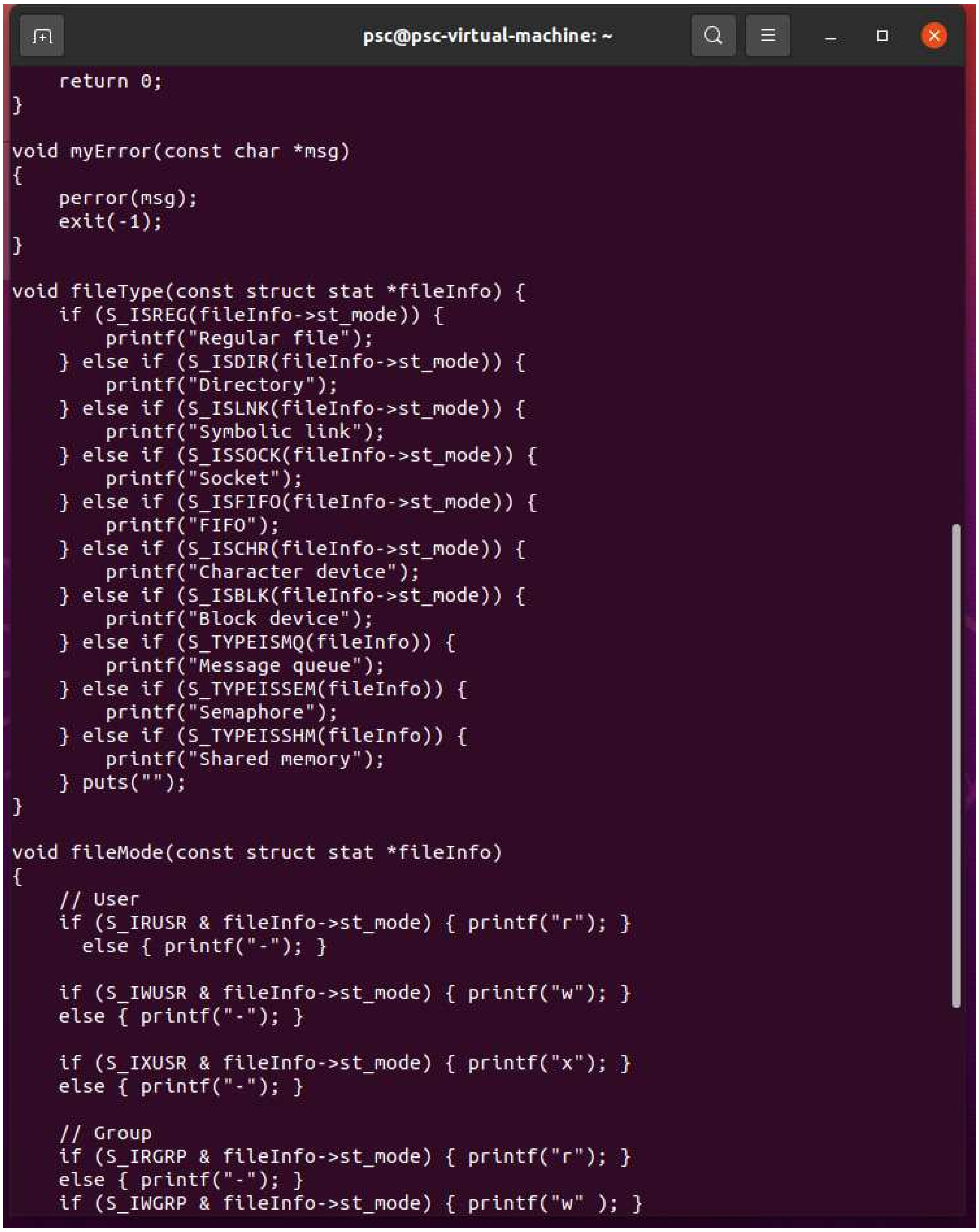


```
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

#define MAX_BUF_SIZE 16
#define PERMS 0644

void myError(const char *msg);

int main(void){
        int fd1 =0;
    char *pathname1="./original.txt";
    struct stat fileinfo;
    char *text;

    int fd2=0;
    char *pathname2="./copy.txt";

    fd1=open(pathname1, O_RDONLY);
    if(fd1==-1){myError("open() error!");}
    if(fstat(fd1, &fileinfo)==-1){myError("fstat() error!");}

    text=(char*)malloc(fileinfo.st_size);
    memset(text,0x00,fileinfo.st_size);
    if(read(fd1,(char*)text,fileinfo.st_size)==-1)
        myError("read() error!");

    fd2=open(pathname2, O_CREAT | O_WRONLY, PERMS);
    if(fd2==-1){myError("open() error!");}

    if(write(fd2,text,fileinfo.st_size)==-1)
        myError("write() error!");

    free(text);
    close(fd2);
    close(fd1);

    return 0;
}

void myError(const char*msg){perror(msg); exit(-1);}
psc@psc-virtual-machine:~$ ./fstat
psc@psc-virtual-machine:~$ ls
chdir     fstat.c     mkdir.c       readdir.c  다운로드  사진
chdir.c   getpwuid    moss.pl       stat       문서      음악
copy.txt  getpwuid.c  original.txt  stat.c     바탕화면  템플릿
fstat     mkdir       readdir       공개       비디오
psc@psc-virtual-machine:~$ 
```

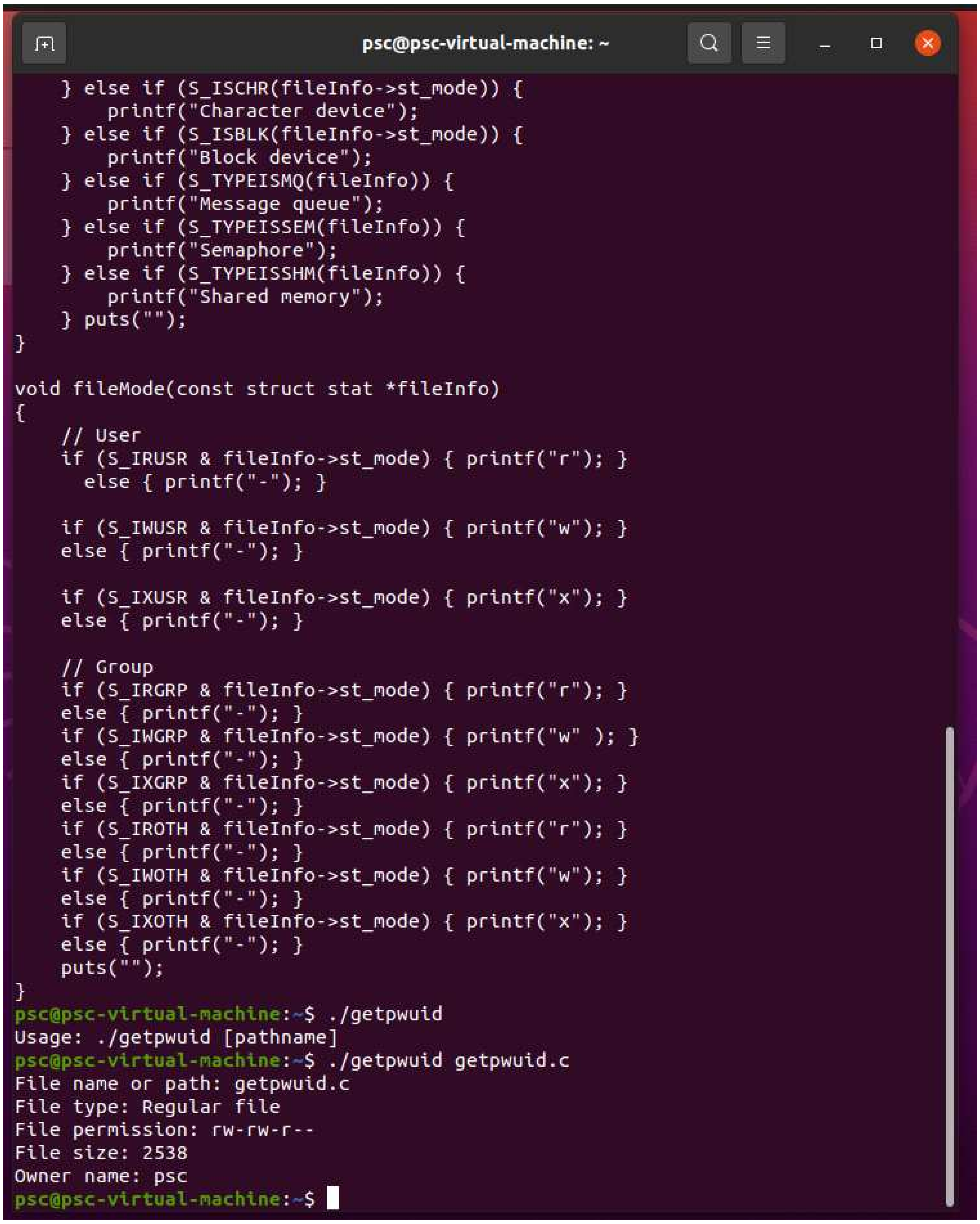


psc@psc-virtual-machine: ~

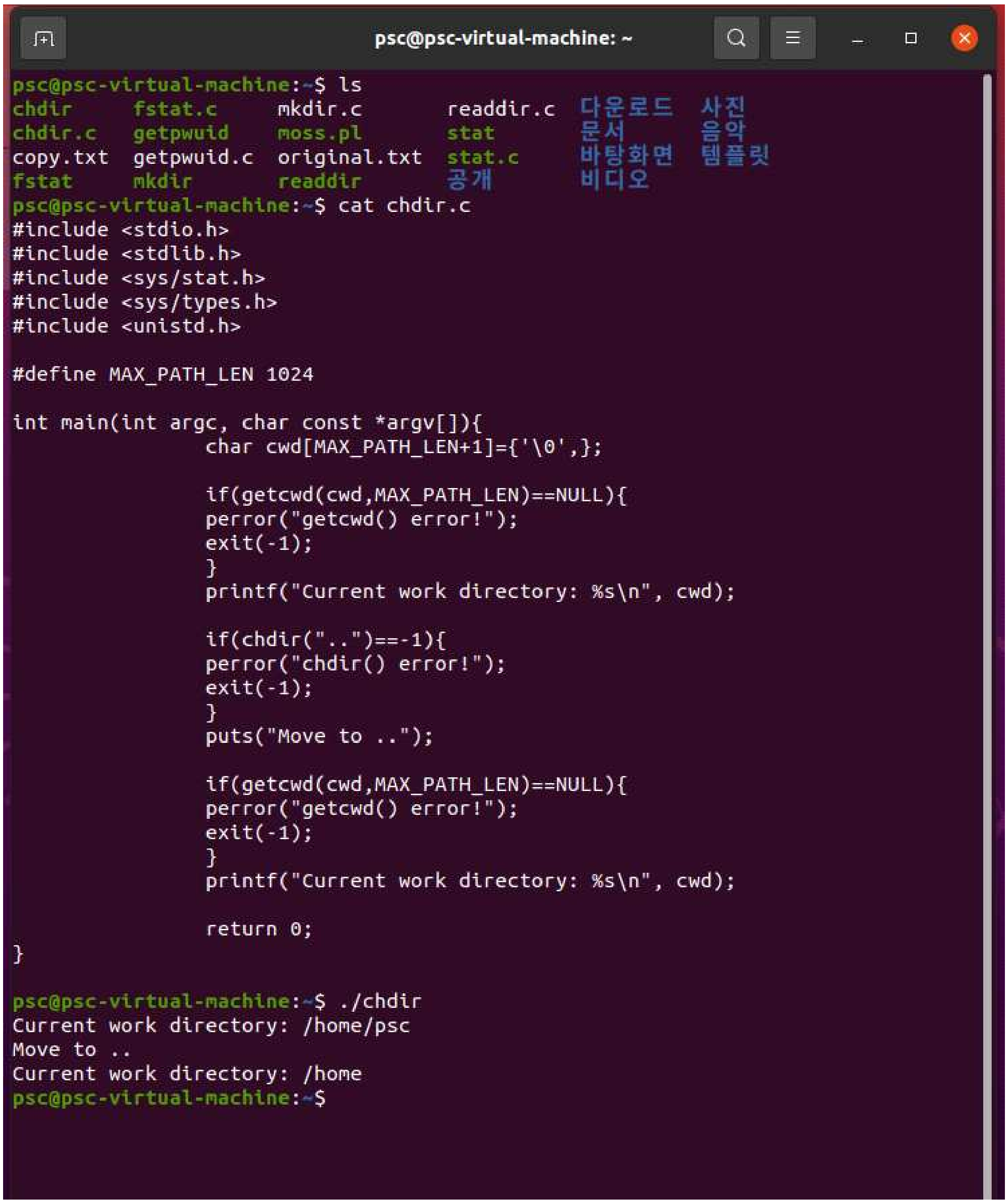
```
psc@psc-virtual-machine:~$ ls
chdir     fstat.c     mkdir.c       readdir.c  다운로드  사진
chdir.c   getpwuid    moss.pl       stat       문서      음악
copy.txt  getpwuid.c  original.txt  stat.c     바탕화면  템플릿
fstat     mkdir       readdir       공개       비디오
psc@psc-virtual-machine:~$ cat getpwuid.c
#include <fcntl.h>
#include <pwd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

void myError(const char*msg);
void fileType(const struct stat *fileinfo);
void fileMode(const struct stat* fileinfo);

int main(int argc, char const*argv[])
{
    struct stat fileinfo;
    struct passwd*userinfo;

    if(argc !=2)
    {
        printf("Usage: %s [pathname]\n", argv[0]);
                exit(-1);
    }

    printf("File name or path: %s\n",argv[1]);
    if(stat(argv[1], &fileinfo)==-1){
        myError("stat() error!");
    }

    printf("File type: ");
    fileType(&fileinfo);

    printf("File permission: ");
    fileMode(&fileinfo);

    printf("File size: %ld\n", fileinfo.st_size);

    userinfo=getpwuid(fileinfo.st_uid);
    printf("Owner name: %s\n", userinfo->pw_name);

    return 0;
}

void myError(const char *msg)
```



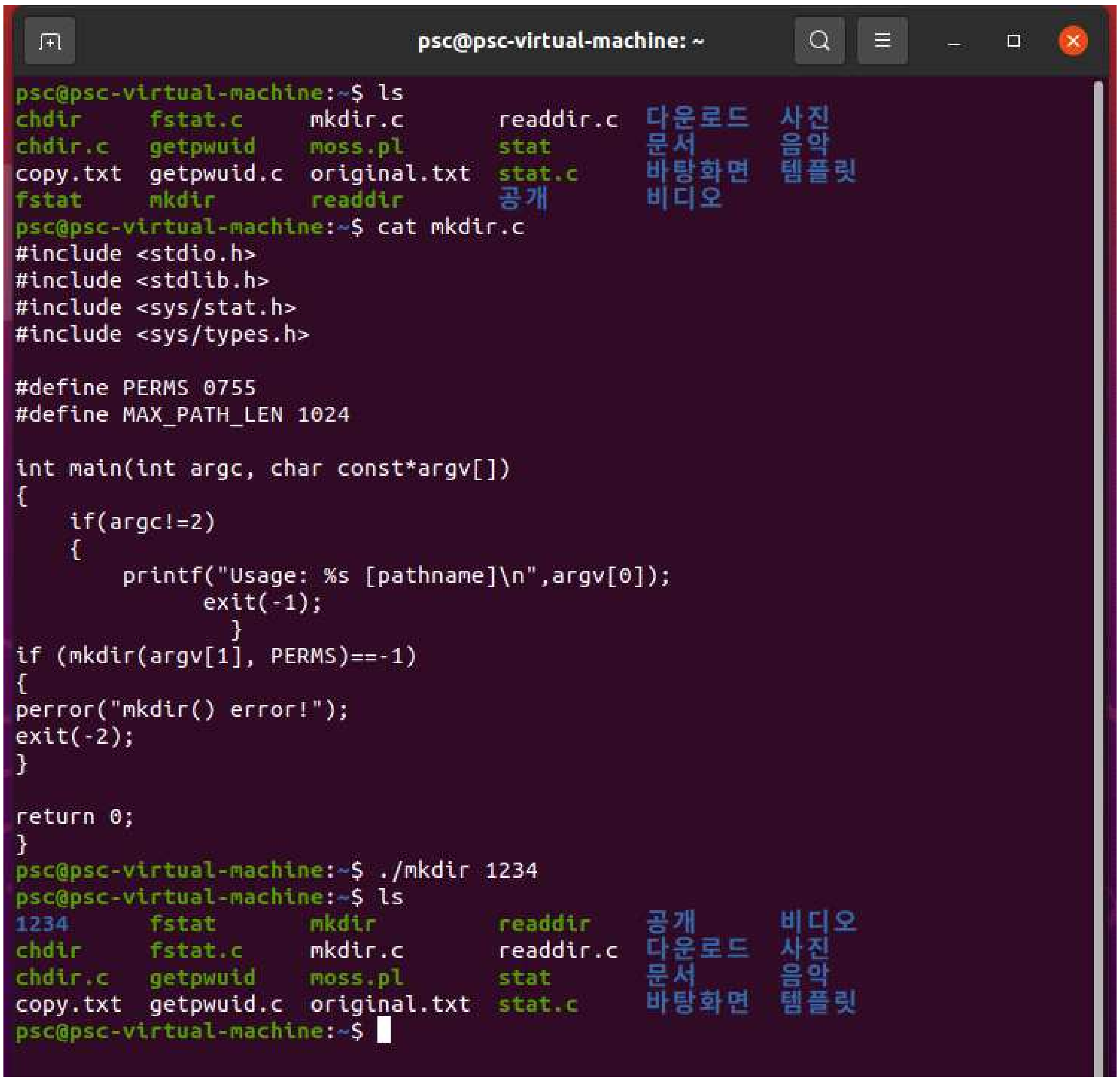
```
    return 0;
}

void myError(const char *msg)
{
    perror(msg);
    exit(-1);
}

void fileType(const struct stat *fileInfo) {
    if (S_ISREG(fileInfo->st_mode)) {
        printf("Regular file");
    } else if (S_ISDIR(fileInfo->st_mode)) {
        printf("Directory");
    } else if (S_ISLNK(fileInfo->st_mode)) {
        printf("Symbolic link");
    } else if (S_ISSOCK(fileInfo->st_mode)) {
        printf("Socket");
    } else if (S_ISFIFO(fileInfo->st_mode)) {
        printf("FIFO");
    } else if (S_ISCHR(fileInfo->st_mode)) {
        printf("Character device");
    } else if (S_ISBLK(fileInfo->st_mode)) {
        printf("Block device");
    } else if (S_TYPEISMQ(fileInfo)) {
        printf("Message queue");
    } else if (S_TYPEISSEM(fileInfo)) {
        printf("Semaphore");
    } else if (S_TYPEISSHM(fileInfo)) {
        printf("Shared memory");
    } puts("");
}

void fileMode(const struct stat *fileInfo)
{
    // User
    if (S_IRUSR & fileInfo->st_mode) { printf("r"); }
      else { printf("-"); }

    if (S_IWUSR & fileInfo->st_mode) { printf("w"); }
    else { printf("-"); }

    if (S_IXUSR & fileInfo->st_mode) { printf("x"); }
    else { printf("-"); }

    // Group
    if (S_IRGRP & fileInfo->st_mode) { printf("r"); }
    else { printf("-"); }
    if (S_IWGRP & fileInfo->st_mode) { printf("w" ); }
```


psc@psc-virtual-machine: ~

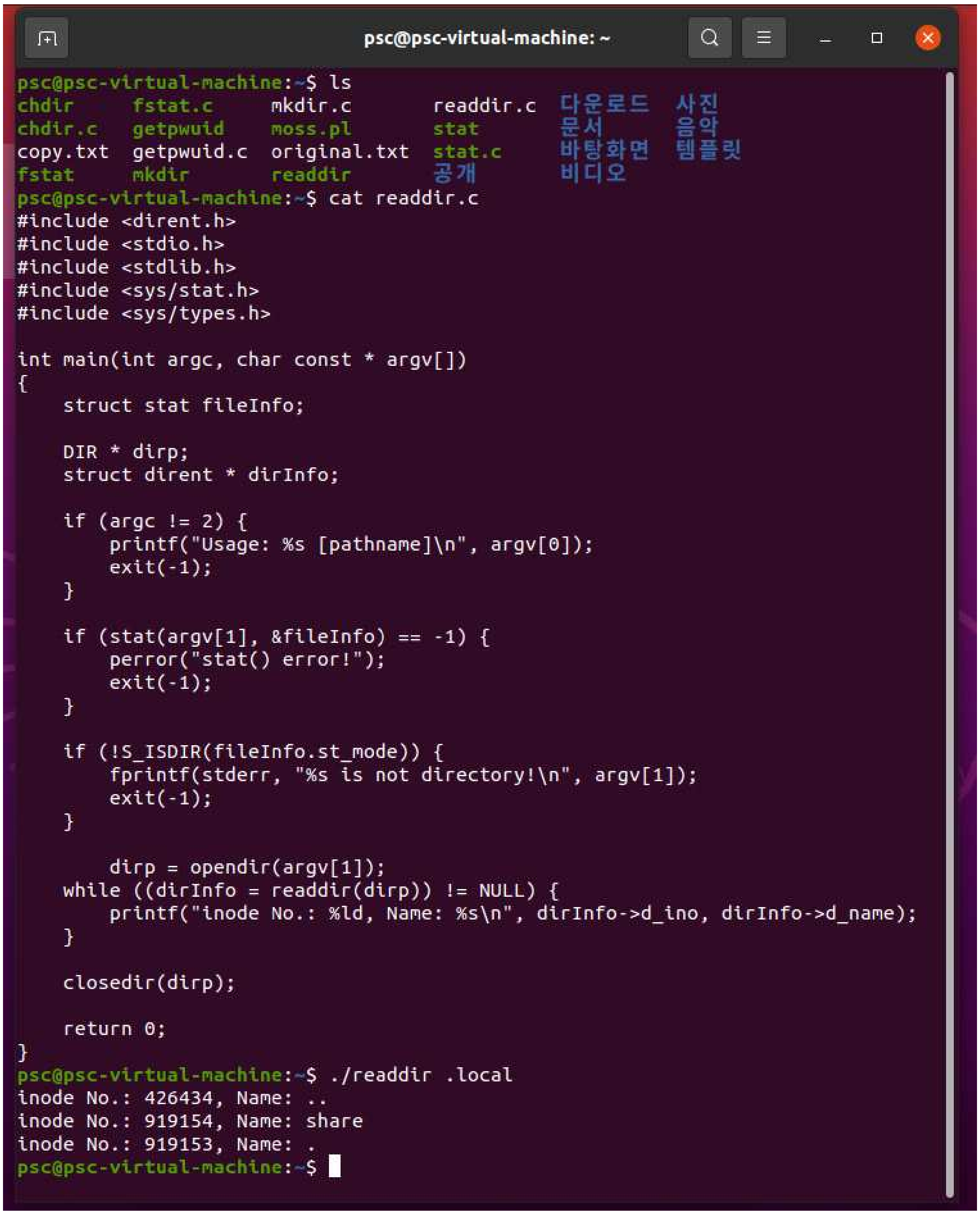
```
    } else if (S_ISCHR(fileInfo->st_mode)) {
        printf("Character device");
    } else if (S_ISBLK(fileInfo->st_mode)) {
        printf("Block device");
    } else if (S_TYPEISMQ(fileInfo)) {
        printf("Message queue");
    } else if (S_TYPEISSEM(fileInfo)) {
        printf("Semaphore");
    } else if (S_TYPEISSHM(fileInfo)) {
        printf("Shared memory");
    } puts("");
}

void fileMode(const struct stat *fileInfo)
{
    // User
    if (S_IRUSR & fileInfo->st_mode) { printf("r"); }
      else { printf("-"); }

    if (S_IWUSR & fileInfo->st_mode) { printf("w"); }
    else { printf("-"); }

    if (S_IXUSR & fileInfo->st_mode) { printf("x"); }
    else { printf("-"); }

    // Group
    if (S_IRGRP & fileInfo->st_mode) { printf("r"); }
    else { printf("-"); }
    if (S_IWGRP & fileInfo->st_mode) { printf("w" ); }
    else { printf("-"); }
    if (S_IXGRP & fileInfo->st_mode) { printf("x"); }
    else { printf("-"); }
    if (S_IROTH & fileInfo->st_mode) { printf("r"); }
    else { printf("-"); }
    if (S_IWOTH & fileInfo->st_mode) { printf("w"); }
    else { printf("-"); }
    if (S_IXOTH & fileInfo->st_mode) { printf("x"); }
    else { printf("-"); }
    puts("");
}
psc@psc-virtual-machine:~$ ./getpwuid
Usage: ./getpwuid [pathname]
psc@psc-virtual-machine:~$ ./getpwuid getpwuid.c
File name or path: getpwuid.c
File type: Regular file
File permission: rw-rw-r--
File size: 2538
Owner name: psc
psc@psc-virtual-machine:~$ 
```



```
psc@psc-virtual-machine:~$ ls
chdir      fstat.c     mkdir.c       readdir.c  다운로드  사진
chdir.c    getpwuid    moss.pl       stat       문서      음악
copy.txt   getpwuid.c  original.txt  stat.c     바탕화면  템플릿
fstat      mkdir       readdir       공개       비디오
psc@psc-virtual-machine:~$ cat chdir.c
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

#define MAX_PATH_LEN 1024

int main(int argc, char const *argv[]){
                char cwd[MAX_PATH_LEN+1]={'\0',};

                if(getcwd(cwd,MAX_PATH_LEN)==NULL){
                perror("getcwd() error!");
                exit(-1);
                }
                printf("Current work directory: %s\n", cwd);

                if(chdir("..")==-1){
                perror("chdir() error!");
                exit(-1);
                }
                puts("Move to ..");

                if(getcwd(cwd,MAX_PATH_LEN)==NULL){
                perror("getcwd() error!");
                exit(-1);
                }
                printf("Current work directory: %s\n", cwd);

                return 0;
}

psc@psc-virtual-machine:~$ ./chdir
Current work directory: /home/psc
Move to ..
Current work directory: /home
psc@psc-virtual-machine:~$
```



```
psc@psc-virtual-machine:~$ ls
chdir     fstat.c     mkdir.c       readdir.c  다운로드  사진
chdir.c   getpwuid    moss.pl       stat       문서      음악
copy.txt  getpwuid.c  original.txt  stat.c     바탕화면  템플릿
fstat     mkdir       readdir       공개       비디오
psc@psc-virtual-machine:~$ cat mkdir.c
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>

#define PERMS 0755
#define MAX_PATH_LEN 1024

int main(int argc, char const*argv[])
{
    if(argc!=2)
    {
        printf("Usage: %s [pathname]\n",argv[0]);
              exit(-1);
                }
if (mkdir(argv[1], PERMS)==-1)
{
perror("mkdir() error!");
exit(-2);
}

return 0;
}
psc@psc-virtual-machine:~$ ./mkdir 1234
psc@psc-virtual-machine:~$ ls
1234      fstat       mkdir         readdir    공개      비디오
chdir     fstat.c     mkdir.c       readdir.c  다운로드  사진
chdir.c   getpwuid    moss.pl       stat       문서      음악
copy.txt  getpwuid.c  original.txt  stat.c     바탕화면  템플릿
psc@psc-virtual-machine:~$ 
```


psc@psc-virtual-machine: ~

```
psc@psc-virtual-machine:~$ ls
chdir      fstat.c     mkdir.c       readdir.c  다운로드  사진
chdir.c    getpwuid    moss.pl       stat       문서      음악
copy.txt   getpwuid.c  original.txt  stat.c     바탕화면  템플릿
fstat      mkdir       readdir       공개       비디오
psc@psc-virtual-machine:~$ cat readdir.c
#include <dirent.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <sys/types.h>

int main(int argc, char const * argv[])
{
    struct stat fileInfo;

    DIR * dirp;
    struct dirent * dirInfo;

    if (argc != 2) {
        printf("Usage: %s [pathname]\n", argv[0]);
        exit(-1);
    }

    if (stat(argv[1], &fileInfo) == -1) {
        perror("stat() error!");
        exit(-1);
    }

    if (!S_ISDIR(fileInfo.st_mode)) {
        fprintf(stderr, "%s is not directory!\n", argv[1]);
        exit(-1);
    }

        dirp = opendir(argv[1]);
    while ((dirInfo = readdir(dirp)) != NULL) {
        printf("inode No.: %ld, Name: %s\n", dirInfo->d_ino, dirInfo->d_name);
    }

    closedir(dirp);

    return 0;
}
psc@psc-virtual-machine:~$ ./readdir .local
inode No.: 426434, Name: ..
inode No.: 919154, Name: share
inode No.: 919153, Name: .
psc@psc-virtual-machine:~$
```