**OS LAB03**

**COMMANDS CODE IN C**

**PWD COMMAND**

#include <unistd.h>

#include <stdio.h>

int main() {

char arr[500];

char\* s = getcwd(arr, sizeof(arr));

if(!s){

printf("Error Occured");

}else{

printf("arr buffer: %s\n", arr);

printf("char pointer: %s\n", s);

}

return 0;

}

**Simple LS COMMAND**

#include <sys/types.h>

#include <dirent.h>

#include <stdio.h>

#include <dirent.h>

int main(int argc, char\* argv[]) {

// DIR \*opendir(const char \*name);

DIR \*d = opendir(argv[1]);

if(!d){

printf("Failure\n");

}else{

struct dirent \*dir = readdir(d);

while(dir){

printf("%s\n", dir->d\_name);

dir = readdir(d);

}

}

return 0;

}

**Recursive LS ls -R COMMAND**

#include <sys/types.h>

#include <dirent.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <sys/stat.h>

#include <unistd.h>

void recursiveLS(DIR \*dirp, char \*path){

struct dirent \*dir = readdir(dirp);

while(dir){

if(dir->d\_name[0] != '.'){

printf("%s\n", dir->d\_name);

// if directory then recursivly print that

if(dir->d\_type == DT\_DIR){

// generate new path

char \*newpath = malloc(strlen(path) + strlen(dir->d\_name) + 2);

strcpy(newpath, path);

strcat(strcat(newpath, "/"), dir->d\_name);

DIR \*newdir = opendir(newpath);

recursiveLS(newdir, newpath);

free(newpath);

}

}

dir = readdir(dirp);

}

}

int main(int argc, char\* argv[]) {

// DIR \*opendir(const char \*name);

DIR \*dirp = opendir(argv[1]);

if(!dirp){

printf("Failure\n");

}else{

recursiveLS(dirp, argv[1]);

}

return 0;

}