#include <sys/param.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

#include <dirent.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <string.h>

void list\_simply\_directory( char \*dir\_path );

void list\_R\_directory( char \*dir\_path );

void Types\_of\_File(char \*dir\_path);

void Current\_Directory\_Address(char \*dir\_path);

char \*get\_current\_dir\_name(void);

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

int number;

if(argc==1){

Current\_Directory\_Address(".");

list\_simply\_directory(".");

return 0;

}else if (argv[1][0]!='-'){

Current\_Directory\_Address(argv[1]);

list\_simply\_directory(argv[1]);

}else if(!strcmp(argv[1],"-FR")){

Types\_of\_File(".");

list\_R\_directory(".");

}else{

while ((number = getopt(argc, argv, ":F:R:")) != -1){

switch (number) {

case 'F':

Types\_of\_File(optarg);

// error use puts(optarg);

break;

case 'R':

list\_R\_directory(optarg);

// error use puts(optarg);

break;

case '?':

printf("arguments error!\n");

break;

case ':':

if(strcmp(argv[1],"-F")==0){

Types\_of\_File(".");

}

if(strcmp(argv[1],"-R")==0){

list\_R\_directory(".");

}

break;

}

}

}

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FUNCTION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

////////////////////////////////////////////list\_simply\_directory/////////////////////////////////////////////////////

void list\_simply\_directory( char \*dir\_path ) {

struct dirent \*\*namelist;

int i, total;

total = scandir(dir\_path, &namelist, NULL, alphasort);

if(total < 0){

perror("scandir");

exit(1);

}else{

for(i = 0; i < total; i++){

if(strcmp(".",namelist[i]->d\_name)==0 || strcmp("..",namelist[i]->d\_name)==0 ) continue ;

printf("%s\n", namelist[i]->d\_name );

}

}

}

////////////////////////////////////////////list\_R\_directory///////////////////////////////////////////////////////

void list\_R\_directory( char \*dir\_path ){

int i,total;

struct stat buf;

struct dirent \*\*namelist;

Current\_Directory\_Address(dir\_path); //or puts(get\_current\_dir\_name());

list\_simply\_directory(dir\_path);

putchar('\n');

total = scandir(dir\_path, &namelist, NULL, alphasort);

if(total < 0){

perror("scandir");

exit(1);

}else{

for(i=0;i<total;i++){

if(strcmp(".",namelist[i]->d\_name)==0 || strcmp("..",namelist[i]->d\_name)==0 ) continue ;

stat(namelist[i]->d\_name,&buf);

if(namelist[i]->d\_type == DT\_DIR){ /\*OR S\_ISDIR(buf.st\_mode)\*/

int number=chdir(dir\_path);

if(number==-1) {

perror("chdir");

exit(1);

}

list\_R\_directory(namelist[i]->d\_name);

if(strcmp(dir\_path,".")!= 0) chdir("..");

}

}

}

}

////////////////////////////////////////////Types\_of\_File///////////////////////////////////////////////////////

void Types\_of\_File(char \*dir\_path){

struct dirent \*\*namelist;

struct stat buf;

int i, total,number;

Current\_Directory\_Address(dir\_path);

total = scandir(dir\_path, &namelist, NULL, alphasort);

for(i = 0; i < total; i++){

chdir(dir\_path);

number=stat(namelist[i]->d\_name,&buf);

if(number==-1){

perror("stat");

exit(1);

}

if(strcmp(dir\_path,".")!= 0) chdir("..");

if(S\_ISDIR(buf.st\_mode)) { //S\_ISDIR(st\_mode)：是否是一個目錄

//remove director . and ..

if(strcmp(".",namelist[i]->d\_name)==0 || strcmp("..",namelist[i]->d\_name)==0 ) continue ;

printf("%s/\n", namelist[i]->d\_name );

}else if(!access(namelist[i]->d\_name,X\_OK)){ //如果是執行檔加上\*

printf("%s\*\n", namelist[i]->d\_name );

}else{ //如果是文件檔直接列出

printf("%s\n", namelist[i]->d\_name );

}

}

}

////////////////////////////////////////////Current\_Directory\_Address///////////////////////////////////////////////////////

void Current\_Directory\_Address(char \*dir\_path){

char \*dir;

int number;

number=chdir(dir\_path);

if(number==-1) {

perror("chdir");

exit(1);

}

long pathmaxlen = pathconf(dir\_path, \_PC\_PATH\_MAX);

dir = getcwd((char \*)NULL, pathmaxlen + 1);

if(dir == NULL) {

perror("getcwd");

exit(1);

}

printf("%s:\n", dir);

free(dir);

if(strcmp(dir\_path,".")!= 0) chdir("..");

}