#include "rmdup.h"

int sortFiles(char\* dir\_path) {

char filepath[PATH\_MAX+1];

DIR \*diretorio;

struct dirent \*d;

struct stat stat;

pid\_t pid;

int status;

if ((diretorio = opendir (dir\_path)) == NULL){ //abre diretorio

perror("Error");

exit(1);

}

while ((d = readdir (diretorio)) != NULL) //lê

{

sprintf(filepath,"%s/%s",dir\_path,d->d\_name);

lstat(filepath,&stat);

if (S\_ISDIR(stat.st\_mode) && strcmp(d->d\_name,".") != 0 && strcmp(d->d\_name,"..") != 0)//verifica se é diretorio

{

sortFiles(filepath);

}

}

pid = fork();

if(pid == 0) //child

{

if(execlp("./lsdir","lsdir",dir\_path,NULL) == -1){

perror("Error");

fprintf(stderr,"Fork error\n");

exit(1);

}

}

else if (pid == -1) //erro

{

fprintf(stderr,"Fork error\n");

exit (2);

}

else if (pid != 0) //father

{

if(wait(&status) != pid)

exit(1);

}

closedir (diretorio);

return 0;

}

//==========================================================================================================================

int sameContent(char\* pathfile1,char\* pathfile2) //returns 0 - yes , -1 - no

{

FILE \*file1, \*file2;

char line1[BUFFER\_SIZE], line2[BUFFER\_SIZE];

file1 = fopen(pathfile1,"r");

file2 = fopen(pathfile2,"r");

if (file1 == NULL || file2 == NULL)

{

perror("Error");

exit(1);

}

while(fgets(line1,BUFFER\_SIZE,file1) && fgets(line2,BUFFER\_SIZE,file2)){

if(strcmp(line1,line2) != 0)

return 1;

}

fclose(file1);

fclose(file2);

return 0;

}

//==========================================================================================================================

int analyseFiles()

{

InfoLine infol , infol2;

FILE \*file, \*file2, \*fileHardLinks;

int i;

char line[BUFFER\_SIZE], line2[BUFFER\_SIZE];

int pos1,pos2, pos3 , pos4; //blank spaces position in line

if((file = fopen("files.txt","r"))==NULL){

perror("Error");

fprintf(stderr, "files.txt not found.\n");

exit(1);}

if((file2 = fopen("files.txt","r"))==NULL){

perror("Error");

fprintf(stderr, "files.txt not found.\n");

exit(1);}

if((fileHardLinks = fopen("hlinks.txt","a+"))==NULL){

perror("Error");

fprintf(stderr, "hlinks.txt not found.\n");

exit(1);}

fgets(line2,BUFFER\_SIZE,file2);

while(fgets(line,BUFFER\_SIZE,file) && fgets(line2,BUFFER\_SIZE,file2)){

pos1 = -1;

pos2 = -1;

pos3 = -1;

pos4 = -1;

for (i=0; i<strlen(line); i++)

{

if (line[i]== '#')

{

if (pos1 == -1)

pos1 = i;

else if(pos2 == -1)

pos2 = i;

else if(pos3 == -1)

pos3 = i;

else{

pos4 = i;

break;

}

}

}

//name

strncpy(infol.name, line, pos1);

infol.name[pos1]='\0';

//data

strncpy(infol.data,line+pos1 + 1, 17);

infol.data[17]='\0';

//permissions

strncpy(infol.permissions, line + pos2 + 1,10);

infol.permissions[10]='\0';

//inode

strncpy(infol.inode,line+pos3 + 1,6);

infol.inode[6]='\0';

//path

strncpy(infol.path, line + pos4 + 1, strlen(line)-pos4-2);

infol.path[strlen(line)-pos4-2]='\0';

if(infol.name[0] == line2[0]){

pos1 = -1;

pos2 = -1;

pos3 = -1;

pos4 = -1;

for (i=0; i<strlen(line2); i++)

{

if (line2[i]== '#')

{

if (pos1 == -1)

pos1 = i;

else if(pos2 == -1)

pos2 = i;

else if(pos3 == -1)

pos3 = i;

else {

pos4 = i;

break;

}

}

}

//name

strncpy(infol2.name, line2, pos1);

infol2.name[pos1]='\0';

//data

strncpy(infol2.data,line2+pos1 + 1, 17);

infol2.data[17]='\0';

//permissions

strncpy(infol2.permissions, line2 + pos2 + 1,10);

infol2.permissions[10]='\0';

//inode

strncpy(infol2.inode,line2+pos3 + 1,6);

infol2.inode[6]='\0';

//path

strncpy(infol2.path, line2 + pos4 + 1, strlen(line2)-pos4-2);

infol2.path[strlen(line2)-pos4-2]='\0';

//mesmo nome, permissões e conteudo

if (strcmp(infol.permissions,infol2.permissions) == 0 && strcmp(infol.name,infol2.name) == 0 && sameContent(infol.path,infol2.path) == 0 && strcmp(infol.inode,infol2.inode) != 0)

{

if(unlink(infol2.path)!=0)

printf("Errno = %d\n",errno);

else{

if(link(infol.path,infol2.path) != 0)

printf("Errno = %d\n",errno);

else

fprintf(fileHardLinks,"%s %s\n",infol2.name,infol2.path);

}

}

}

}

fclose(file2);

fclose(file);

fclose(fileHardLinks);

return 0;

}

//==========================================================================================================================

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

int file, fileHL;

if (argc != 2){

printf("WRONG NUMBER OF ARGUMENTS\n");

exit(1);

}

//Clean the file

if((file = open("files.txt",O\_WRONLY | O\_CREAT | O\_TRUNC,0644)) == -1){ //clean sorted files

perror("Error");

fprintf(stderr,"Couldn't open files.txt.\n");

exit(1);

}

close(file);

if((file = open("tmp\_files.txt",O\_WRONLY | O\_CREAT | O\_TRUNC,0644)) == -1){ //clean temp files

perror("Error");

fprintf(stderr,"Couldn't open tmp\_files.txt.\n");

exit(1);

}

close(file);

if((fileHL = open("hlinks.txt",O\_WRONLY | O\_CREAT | O\_SYNC | O\_TRUNC,0644)) == -1){ //clean hardlinks

perror("Error");

fprintf(stderr,"Couldn't open hlinks.txt.\n");

exit(1);

}

close(fileHL);

//modify files.txt with the info from the directory and all subdirectorys

sortFiles(argv[1]);

analyseFiles();

char currentdirectory[1024];

getcwd(currentdirectory,sizeof(currentdirectory));

char tmp\_path[1024];

sprintf(tmp\_path,"%s/tmp\_files.txt",currentdirectory);

unlink(tmp\_path);

return 0;

}