Recitation 4

TA Hanxiong Chen
hc691@rutgers.edu

Assignment related

Opendir()

- This is a function to open a directory
- It only accesses to directory, not file content
- Need to include <dirent.h>

Assignment related (cont’d)

readdir()

- This is the function to read directory info
- The info will be stored in dirent struct
- We can only read inode info
- If nothing to read in the directory, the function returns NULL

Assignment related (cont'd)

What's in dirent struct?

```
struct dirent {
        ino_t       d_ino;     /* Inode number */
        off_t       d_off;     /* Not an offset; see below */
        unsigned short d_reclen;    /* Length of this record */
        unsigned char  d_type;      /* Type of file; not supported
                                       by all filesystem types */
        char        d_name[256]; /* Null-terminated filename */
    };
```

Assignment related (cont’d)

Example:

```
#include <stdio.h>
#include <dirent.h>

int main (int c, char *v[]) {
int len;
struct dirent *pDirent;
DIR *pDir;
if (c < 2) {
printf ("Usage: testprog <dirname>\n");
return 1;
}
pDir = opendir (v[1]);
if (pDir == NULL) {
printf ("Cannot open directory '%s'\n", v[1]);
return 1;     //not a good way
}
while ((pDirent = readdir(pDir)) != NULL) {
printf ("[%s]\n", pDirent->d_name);
}
closedir (pDir);
return 0;
}
```

Assignment related (cont’d)

- You can treat this process as a recursive method to do depth first search
- Create two modules: one for directory and one for file
  - If there is a directory, do opendir() and readdir()
  - If there is a file, do fdopen()/open() and fread()/read()

HW 5: Implement ls

0. Using opendir and readdir, open the current directory and output all filenames until there are no more char * base = "./";

```
  DIR * thingy = opendir(base);

  dirent * newfile = readdir(thingy);
```

1. Parse the dirent struct to see if an entry is a directory or a file. If it is a file, prepend "./" to the filename, if it is a directory, don't.

```
  .. if newfile != NULL

  //check type field of newfile dirent struct to determine the type of this file endpoint

  newfile->d_type

// compare with system defines for different endpoint types (3rd paragraph under 'NOTES' in man 3 readdir).

  ... if == DT_REG //regular file

  elseif == DT_DIR //directory

....
```

HW 5 (cont.)

2. Open a file handle to each file and use lseek to determine the file's size in bytes, and print out the file's size next to its name.

```
//assemble name of file using base directory and current path/name // concatenate all path up until now…
        strcat(newpath, base)
// add currend name if it is a file…
        newerpath = realloc(newpath, strlen(newpath)+strlen(newfile->d_name));
// d_name is REQUIRED to have a terminating null byte by standard ... yippee!
        strcat(newerpath, newfile->d_name);
        int checkFD = open(newerpath, RD_ONLY);
 ... if no error…
        int len = lseek(checkFD, 0, SEEK_END);
        close(checkFD);
printf( filename with full path, either color to indicate file/dir or put a "/" at the end to indicate dir, and number of bytes of size, if a file )
//be sure to closedir() when done with dir descriptor
```


TIME FOR REVIEW


???