مقدمه

در این تمرین قرار است برنامه‌ای بنویسید که سیستم فایل‌ها را مشابه آنچه به صورت روزانه از آن استفاده می‌کنید پیاده‌سازی کند. برنامه‌ای که شما خواهید نوشت دستوراتی مانند ایجاد فایل یا پوشه، کپی، و حذف را از کاربر دریافت نموده و عملیات متناظر با آن‌ها را انجام می‌دهد.

سیستم فایل‌ها

سیستم فایل‌ها (File System) برنامه‌ای است که مدیریت اطلاعات ذخیره‌شده روی هارددیسک، و سایر دستگاه‌های ذخیره‌ی اطلاعات را انجام می‌دهد. مثلاً وقتی شما یک فایل جدید ایجاد می‌کنید، سیستم عامل از سیستم فایل‌ها می‌خواهد تا آن فایل را در محل مورد نظر ایجاد کند. البته بر خلاف یک سیستم فایل‌های واقعی که اطلاعات خود را در دیسک ذخیره می‌کند، شما همه‌ی اطلاعات را در حافظه نگهداری می‌کنید[1].

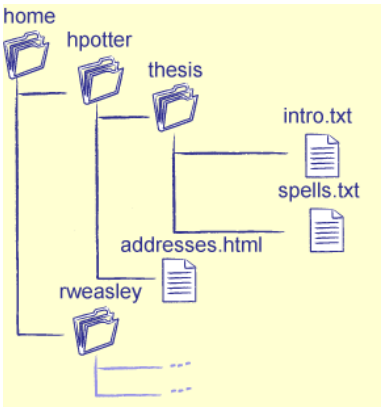
برای اینکه بتوانید رفتار سیستم فایل‌ها را شبیه‌سازی کنید لازم است با نحوه‌ی ذخیره‌ی اطلاعات در دیسک‌ها بیشتر آشنا شوید. در این قسمت توضیح مختصر (و ساده‌شده[2]ای) را در این خصوص بیان می‌کنیم.

ذخیره‌ی اطلاعات در دیسک

فضای قابل استفاده در یک دیسک به قسمت‌هایی با ظرفیت یکسان تقسیم می‌شود. به این قسمت‌ها سکتور (Sector) گفته می‌شود. در این تمرین ظرفیت یک سکتور را ۶۴ بایت در نظر می‌گیریم (البته در دیسک‌های واقعی ظرفیت سکتورها بیشتر است). سکتور واحد ذخیره و بازیابی اطلاعات در یک دیسک است. این به این معنی است که در هر بار خواندن یا نوشتن اطلاعات در دیسک دقیقاً ۶۴ بایت داده منتقل می‌شود. مثلاً اطلاعات یک فایل ۱ کیلوبایتی در ۱۶ مرحله از دیسک خوانده می‌شود. به همین ترتیب اگر بخواهیم فقط یک بایت از یک فایل را تغییر دهیم، ناچاریم کل بایت‌های سکتور را بخوانیم، بایت مورد نظر را در حافظه تغییر دهیم، و کل ۶۴ بایت را در سکتور مربوطه بنویسیم.

برای ذخیره‌ی هر فایل، تعدادی سکتور به آن فایل اختصاص داده می‌شود. به وضوح بهتر است که سکتورهای اختصاص‌داده‌شده در مجاورت هم باشند (تا سرعت بازیابی اطلاعات یک فایل افزایش پیدا کند)، اما همیشه این کار ممکن نیست. در نتیجه لازم است که سیستم اطلاعات سکتورهای مختلف دیسک را داشته‌باشد. مثلاً اینکه کدام سکتورها خالی هستند، یا هر سکتور به کدام فایل اختصاص داده شده.

همان طور که می‌دانیم، فایل‌ها به صورت دسته‌بندی‌شده در پوشه‌ها ذخیره می‌شوند. با اختصاص هر فایل و پوشه، به پوشه‌ای که در آن قرار گرفته‌است، یک ساختار درختی مشابه شکل زیر پدید می‌آید:



[1] این باعث می‌شود که برنامه‌ی شما یک سیستم فایل‌های مجازی باشد، که هنگام خروج از برنامه، اطلاعات آن از بین می‌رود!
[2] برای ساده‌سازی، از مفاهیم پیچیده‌تری مانند پارتیشن صرف‌نظر می‌کنیم.

به این ساختار، ساختار فایل‌ها گفته می‌شود که نگهداری آن به عهده‌ی سیستم فایل‌ها است. توجه داشته‌باشید که این ساختار فایل‌ها فقط از دیدگاه کاربر وجود دارد، و در سطح دیسک معنایی ندارد. مثلاً در صورتی که شما یک فایل را از یک پوشه به پوشه‌ی دیگر منتقل کنید، عملاً خود فایل در دیسک تغییری نمی‌کند و انتقال فایل تنها در ساختمان‌داده‌های سیستم فایل‌ها تغییر ایجاد می‌کند. به همین دلیل است که عملیات انتقال فایل درون یک دیسک[۳] معمولاً خیلی سریع‌تر از عملیات کپی مشابه آن است.

با توجه به آنچه که گفته‌شد، سیستم فایل‌هایی که شما طراحی می‌کنید باید این اطلاعات را نگهداری کند:

- ساختار درختی فایل‌ها.
- اطلاعات فایل، شامل نام، حجم، متحویات، سکتورهای اختصاص‌داده‌شده و ...
- اطلاعات سکتورها، اینکه هر سکتور خالی است یا نه، و در صورت خالی نبودن، به کدام فایل اختصاص پیدا کرده‌است.

به این ترتیب، برنامه‌ی شما به این صورت کار می‌کند، که هنگام اجرا ظرفیت دیسک را از ورودی دریافت کرده و یک دیسک مجازی در حافظه، با ظرفیت داده‌شده ایجاد می‌کند. سپس یک خط فرمان (Command Line) در اختیار کاربر قرار می‌دهد که دستورات زیر را دریافت نموده و عملیات متناظر با آن را انجام می‌دهد.

دستوراتی که خط فرمان دریافت می‌کند، یک یا دو آدرس را به عنوان ورودی دریافت می‌کنند. آدرس‌ها نشان‌دهنده‌ی فایل‌ها و پوشه‌ها هستند، به این صورت که هر آدرس یک پرونده یا پوشه را به صورت یکتا نشان می‌دهد. در حقیقت آدرس یک فایل یا پوشه، نشان‌دهنده‌ی نام و محل قرارگیری پوشه در ساختار فایل‌ها است.

آدرس می‌تواند به یکی از دو شکل مطلق یا نسبی داده‌شود. آدرس‌های مطلق با کاراکتر / (که نشان‌دهنده‌ی ریشه‌ی درخت ساختار فایل‌ها است) شروع می‌شوند و مستقل از پوشه‌ی جاری هستند. مثلاً /home/hpotter/addresses.html. آدرس‌های نسبی، محل قرارگیری فایل یا پوشه را نسبت به پوشه‌ی جاری نشان می‌دهند، مثلاً اگر پوشه‌ی جاری home/hpotter/ باشد، آدرس نسبی thesis/spells.txt فایل با آدرس مطلق home/hpotter/thesis/spells.txt/ را نشان می‌دهد. به طور خاص، عبارت .. آدرس نسبی پوشه‌ای است که پوشه‌ی جاری در آن قرار دارد (به استثناء ریشه). در پوشه‌ی home/hpotter/، آدرس نسبی rweasley/.. معادل آدرس مطلق home/rweasley/ خواهد بود.

در ادامه، دستورات مختلفی را که خط فرمان شما باید آنها را اجرا کند، مشاهده می‌کنید. در این تعریف‌ها، عباراتی که داخل <> قرار گرفته‌اند پارامترهای اجباری هستند و پارامترهایی که داخل [] قرار گرفته‌اند اختیاری.

- pwd: نمایش آدرس پوشه‌ی جاری.
- cd <DIRECTORY_ADDRESS>: تغییر پوشه‌ی جاری به پوشه‌ی مشخص‌شده.
- ls [DIRECTORY_ADDRESS]: نمایش محتویات پوشه‌ی داده‌شده. اگر پوشه‌ای مشخص نشود، این دستور باید محتویات پوشه‌ی جاری را نمایش دهد. محتویات پوشه باید به ترتیب حروف الفبا و هر یک در یک خط نشان داده‌شوند.
- ls <FILE_ADDRESS>: نمایش اطلاعات فایل، شامل نام، آدرس و حجم فایل.
- rm <FILE_ADDRESS>: حذف فایل داده‌شده.
- rm <DIRECTORY_ADDRESS>: حذف پوشه‌ی داده‌شده به همراه کلیه‌ی محتویات آن.
- mkdir <DIRECTORY_ADDRESS>: ایجاد پوشه با آدرس داده‌شده.
- cp <FILE_ADDRESS> <DIRECTORY_ADDRESS>: کپی فایل داده‌شده به داخل پوشه‌ی داده‌شده.
- cp <FILE_ADDRESS> <FILE_ADDRESS>: کپی فایل داده‌شده به آدرس داده‌شده.
- cp <DIRECTORY_ADDRESS> <DIRECTORY_ADDRESS>: کپی پوشه‌ی اول، به همراه کلیه‌ی محتویات آن، به داخل پوشه‌ی دوم.
- mv <FILE_ADDRESS> <DIRECTORY_ADDRESS>: انتقال فایل داده‌شده به داخل پوشه‌ی داده‌شده.
- mv <FILE_ADDRESS> <FILE_ADDRESS>: انتقال فایل داده‌شده به آدرس داده‌شده (می‌تواند برای تغییر نام فایل استفاده شود).
- mv <DIRECTORY_ADDRESS> <DIRECTORY_ADDRESS>: انتقال پوشه‌ی اول، به همراه کلیه‌ی محتویات آن، به داخل پوشه‌ی دوم (می‌تواند برای تغییر نام پوشه استفاده شود).

[۳] و یک پارتیشن

- put <FILE_NAME> <DIRECTORY_ADDRESS>: خواندن یک فایل واقعی در سیستم عامل و ایجاد یک فایل با همان نام در پوشه‌ی جاری.
- get <FILE_ADDRESS>: خواندن فایل داده‌شده و ایجاد یک فایل واقعی با همان نام در سیستم عامل.
- defrag: مرتب کردن فایل‌ها بر روی دیسک، به نحوی که اولاً سکتورهای هر فایل پشت سر هم باشند، و ثانیاً فضای اختصاص‌یافته به فایل‌ها یکپارچه باشد. این به این معنی است که پس از اجرای این دستور همه‌ی سکتورهای خالی در انتهای دیسک و همه‌ی سکتورهای اختصاص‌داده‌شده در ابتدای دیسک قرار خواهند داشت.

در نوشتن برنامه‌ی خود به این نکات توجه کنید:

- فرض کنید کلیه‌ی فایل‌ها فایل‌های متنی هستند.
- فرض کنید نام کلیه‌ی فایل‌ها و پوشه‌ها فقط از حروف الفبا، ارقام و کاراکتر _ تشکیل شده‌است.
- سیستم فایل‌ها به بزرگی و کوچکی حروف حساس است.
- هنگام اجرای کلیه‌ی فرمان‌ها، باید در صورت بروز خطا، خطا را به کاربر گزارش نموده و سپس خط فرمان را نمایش دهید. برای این کار باید از Exception استفاده کنید.
-

نحوه‌ی تحویل

قبل از تحویل حضوری؛ در موعدی که تعیین خواهد شد به ایمیل زیر ارسال کنید

cpp1395+projectFS@gmail.com

دقت کنید

- طراحی مناسب و پیاده‌سازی کامل و صحیح نیازمندی‌های مسأله نمره‌ی شما در این تمرین را تشکیل خواهند داد. تحویل تمرین به صورت حضوری خواهد بود.
- شما باید تمامی نکات برنامه‌نویسی که در طول این ترم در درس برنامه‌سازی پیشرفته آموخته‌اید را در این تمرین رعایت کنید!