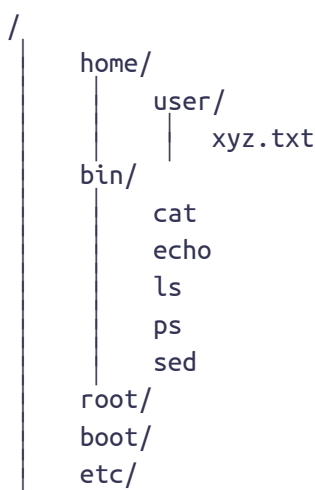


جلسه اول — آشنایی با پوسته

در این جلسه با مقدمات استفاده از رابط پوسته^۱ برای مدیریت فایل‌ها در محیط‌های مشابه یونیکس^۲ آشنا خواهید شد.

ساختار فایل سیستم

فایل سیستم در یونیکس دارای یک ساختار درختی است که رأس‌های میانی آن شافه^۳ هستند. این درخت از ریشه (که با علامت «/» نشان داده می‌شود) شروع می‌شود. شکل زیر ساختار درختی یک فایل سیستم نمونه را نشان می‌دهد.



با پیمودن مسیر شروع شده از ریشه به فایل‌ها و شافه‌ها در این درخت، آدرس آنها تعیین می‌شود: شافه‌های این مسیر از ریشه از چپ به راست کنار هم قرار داده می‌شوند و با علامت «/» جدا می‌گردند. برای نمونه، آدرس فایل «ls» در شکل قبل «/bin/ls» می‌باشد. هر شافه در فایل سیستم دو زیر شافه‌ی مجازی دارد: «.» به همان شافه و «..» به شافه‌ی بالاتر از آن شافه اشاره می‌کند. بنابراین دو آدرس «/root/» و «/bin/../root/» به یک شافه اشاره می‌کنند.

به هر پرده (از جمله پوسته) در سیستم عامل شافه‌ای به نام شافه‌ی جاری^۴ اختصاص داده می‌شود؛ این شافه، آدرسی که پرده در آن در حال اجرا است را مشخص می‌کند. با توجه به این شافه، آدرس فایل‌ها و شافه‌ها به صورت نسبی بیان می‌گردد؛ آدرس‌های نسبی آدرس‌هایی هستند که با «/» شروع نمی‌شوند. برای یافتن مقصد این آدرس‌ها، آدرس شافه‌ی جاری به ابتدای آنها اضافه می‌گردد. به عنوان مثال، در صورتی که شافه‌ی جاری «/home/user/» باشد، آدرس نسبی «./» به شافه‌ی «/home/» اشاره می‌کند و آدرس نسبی «xyz.txt» به فایل «/home/user/xyz.txt» اشاره می‌کند.

1 Shell
2 Unix like
3 Directory
4 Current working directory

اجرای دستورات در پوسته

اصلی‌ترین رابط کاربر در یونیکس برنامه‌ای به نام پوسته می‌باشد. پوسته دستورات را یکی پس از دیگری از ورودی می‌خواند و اجرا می‌کند. برقی از دستورات ابتدایی پوسته و محیط یونیکس در شکل زیر نشان داده شده‌اند.

\$ pwd	#	آدرس شافه‌ی جاری پوسته را چاپ می‌کند
\$ cd path	#	شافه‌ی جاری پوسته را به «path» تغییر می‌دهد
\$ cd ~	#	شافه‌ی جاری پوسته را به شافه‌ی خانه تغییر می‌دهد
\$ cd	#	معادل دستور قبلی
\$ ls	#	همه‌ی فایل‌ها و شافه‌های شافه‌ی جاری را فهرست می‌کند
\$ find -name "pat"	#	مستجمعی همه‌ی فایل‌هایی که نامشان با الگوی «pat» مطابقت می‌کند
\$ find path -name "pat"	#	مشابه دستور قبل برای شافه‌ی «path»
\$ mkdir XYZ	#	شافه‌ای با نام «XYZ» ایجاد می‌کند
\$ rmdir XYZ	#	شافه‌ی «XYZ» را حذف می‌کند؛ شافه باید خالی باشد
\$ rm XYZ	#	فایل «XYZ» را حذف می‌کند
\$ rm -r XYZ	#	به صورت بازگشتی فایل یا شافه‌ی «XYZ» را حذف می‌کند
\$ cp src dir1/	#	فایل مبدأ (پارامتر اول) را به شافه‌ی مقصد (پارامتر دوم) کپی می‌کند
\$ cp -r dir1/ dir2/	#	فایل یا شافه‌ی مبدأ را به صورت بازگشتی کپی می‌کند
\$ mv dir1 dir3	#	فایل یا شافه‌ی اول را به شافه‌ی دوم انتقال می‌دهد
\$ echo "Hello!"	#	عبارت «Hello!» را چاپ می‌کند
\$ cat TEST	#	ممتویات فایل «TEST» را چاپ می‌کند
\$ passwd	#	گذرواژه‌ی کاربر را عوض می‌کند
\$ ls --help	#	برقی از امکانات دستور «ls» را فهرست می‌کند
\$ man ls	#	صفحه‌ی راهنمای دستور داده شده را نشان می‌دهد
\$ date	#	زمان و تاریخ را چاپ می‌کند
\$ ps aux	#	فهرست همه‌ی پردازش‌ها را نشان می‌دهد
\$ pstree	#	درخت پردازش‌ها را نشان می‌دهد
\$ sleep 5	#	پنج ثانیه متوقف می‌ماند

دستورات را در پوسته می‌توان به شکل‌های گوناگونی ترکیب نمود. در ادامه چند مثال برای ترکیب

دستورها نشان داده شده‌اند.

\$ cmd1; cmd2	#	الف) ابتدا دستور «cmd1» و سپس «cmd2» را اجرا می‌کند.
\$ cmd2 `cmd1`	#	ب) عبارت «`cmd1`» را با فریمی دستور «cmd1» جایگزین و دستور «cmd2» را اجرا می‌کند.
\$ cmd1 && cmd2	#	ج) دستور «cmd2» را تنها در صورت موفقیت‌آمیز بودن «cmd1» اجرا می‌کند.
\$ cmd1 cmd2	#	د) دستور «cmd2» را تنها در صورت ناموفقیت‌آمیز بودن «cmd1» اجرا می‌کند.

در پایان هر دستور در پوسته مانند برقی از زبان‌های برنامه‌نویسی می‌توان علامت «;» را قرار داد. در صورتی

که دو دستور مستقل در یک خط بیان کردند می‌توان آنها را با این علامت جدا ساخت (قسمت «الف» از شکل قبل). همچنین، پوسته قبل از اجرای یک دستور، عبارت‌های داخل دو علامت «`» را اجرا می‌کند و آنها را با فروجه‌شان جایگزین می‌نماید (قسمت «ب»). در قسمت‌های «ج» و «د» از شکل قبل دو دستور به صورت شرطی با هم ترکیب می‌گردند: موفقیت یک دستور با توجه به کد برگشتی^۱ آن دستور تعیین می‌گردد: به صورت قراردادی در صورتی که یک برنامه مقدار صفر را به عنوان کد برگشتی برگرداند موفقیت آمیز بوده است و در غیر این صورت مشکلی در اجرای برنامه بوجود آمده است. برای مثال دستور «ls» در صورتی که به آن یک آدرس غیر موجود به عنوان ورودی داده شود، مقداری غیر صفر بر می‌گرداند.

گسترش‌ها در پوسته

یکی از ویژگی‌های پوسته که مشخص کردن تعداد زیادی فایل را آسان می‌کند، ویژگی گسترش نام فایل^۲ در آن می‌باشد. پوسته عبارت‌های شامل علامت‌های «?»، «*» یا «[.]» را به عنوان الگوی فایل‌ها می‌پذیرد و آن عبارت را با فهرست فایل‌هایی که با آنها مطابقت دارند جایگزین می‌کند. در این الگوها، «?» با هر حرفی، «*» با هر رشته‌ای و «[.]» با هر یک از حروف مشخص شده در آن مطابقت می‌کنند. برای مثال «[hc].*» با نام‌های همگی فایل‌های شش‌ه‌ای جاری که پسوند «.c» یا «.h» دارند جایگزین می‌گردد. چند نمونه از گسترش فایل‌ها در پوسته در ادامه نشان داده شده‌اند.

\$ ls	#	فهرست همگی فایل‌ها
a1.c a1.h a2.c a3.c b1.c b1.h b2.c b3.c c1.c c1.h		
\$ ls a*	#	فهرست فایل‌هایی که با «a» شروع می‌شوند
a1.c a1.h a2.c a3.c		
\$ ls *.h	#	فهرست فایل‌های با پسوند «.h»
a1.h b1.h c1.h		
\$ ls a1.[hc]	#	فهرست فایل‌هایی که با الگوی «a1.[hc]» مطابقت دارند
a1.c a1.h		
\$ ls c??.?	#	فهرست فایل‌هایی که با الگوی «c??.?» مطابقت دارند
c1.c c1.h		
\$ ls ?1.c	#	فهرست فایل‌هایی که با الگوی «?1.c» مطابقت دارند
a1.c b1.c c1.c		

علاوه بر گسترش نام فایل‌ها، پوسته عبارت‌های دیگری را نیز گسترش می‌دهد. نام‌های پس از علامت

«\$» با مقدار متغیر پوسته یا مقدار متغیر ممیطی با آن نام جایگزین می‌گردند.

1 Return code
2 File name expansion

\$ echo \$VAR	#	مقدار متغیر مملی یا متغیر پوستهی «VAR» را چاپ می‌کند
\$ echo \${VAR}	#	مشابه دستور قبل
\$ VAR="abc"	#	رشتهی «abc» را به متغیر «VAR» نسبت می‌دهد

صفه‌ی راهنمای دستورات

بیشتر دستورات در یونیکس با گرفتن پارامتر ورودی «-h» یا «--help» فهرستی از ویژگی‌هایشان و چگونگی فعال‌سازی آنها را چاپ می‌کنند. علاوه بر آن، به همراه بیشتر دستورات در محیط یونیکس یک صفه‌ی راهنما وجود دارد که در مورد شیوه‌ی استفاده از آن دستور و امکانات آن توضیح می‌دهد. دستور «man» صفه‌ی راهنمای یک دستور را نمایش می‌دهد.

\$ man ls	#	صفه‌ی راهنمای دستور «ls»؛ برای خروج دکمه‌ی «q» را فشار دهید.
-----------	---	--

تمرین یک

پس از دریافت فایل فشرده‌ی «git-2.6.0.tar.gz»، محتویات آن را با دستور زیر در شافه‌ی «/git-2.6.0» در شافه‌ی فانه‌ی خود باز (Extract) نمایید:

```
$ tar xzvf git-2.6.0.tar.gz
```

سپس شافه‌ای با نام ex1 در شافه‌ی فانه‌ی خود بسازید که سافت‌اری مانند شکل زیر داشته باشند. دقت نمایید که فایل‌های این درخت را باید از فایل‌های باز شده در /git-2.6.0 بگیرید.

```
git/
|
|   include/
|   |
|   |   diff.h
|   |   khash.h
|   |   refs.h
|   |   tar.h
|   |   url.h
|   |   utf8.h
|   |
|   |   src/
|   |   |
|   |   |   diff.c
|   |   |   pager.c
|   |   |   refs.c
|   |   |   url.c
|   |   |   utf8.c
|   |   |
|   |   |   contacts/
|   |   |   |
|   |   |   |   Makefile
|   |   |   |   git-contacts
|   |   |   |   git-contacts.txt
|   |   |
|   |   |   git-log.sh
|   |   |   git-clean.sh
```

گام‌های پیشنهادی برای انجام این تمرین:

- ۱ دریافت و باز کردن فایل «git-2.6.0.tar.gz»
- ۲ ساختن شافه‌های مورد نیاز در «/ex1»
- ۳ یافتن فایل‌های مورد نیاز در شافه‌ی «~/git-2.6.0/»
- ۴ کپی کردن فایل‌های مورد نیاز از شافه‌ی «~/git-2.6.0/»