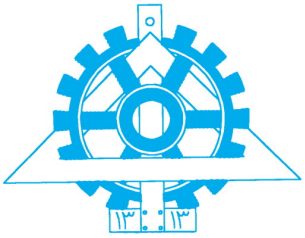

به‌نام‌خدا

سیستم‌عامل پیشرفته - بهار ۱۴۰۳

تمرین اول: آشنایی با اسکریپت‌نویسی در محیط لینوکس

استاد درس: دکتر ناصر یزدانی

دستیار آموزشی: حسین افکار

ایمیل: hossein.afkar@ut.ac.ir

مقدمه

هدف از انجام این تمرین آشنایی با اسکریپت‌نویسی در محیط سیستم‌عامل لینوکس و بر پایه یونیکس است. اولین رابط کاربری سیستم‌عامل با شل سیستم‌عامل شروع می‌شود که در این رابط دستورات به صورت خط به خط (یا بر پایه قوائد زبانی شل موجود) خوانده شده و اجرا می‌شوند.
در این تمرین با ما شل سیستم‌عامل لینوکس یا bash کار می‌کنیم و در حین این تمرین توانایی شما برای نوشتن اسکریپت‌های ساده سنجیده خواهد شد. شل‌ها در محیط سیستم‌عامل کاربردهای فرآوانی دارند که مثال‌هایی از آن می‌تواند استفاده در سیستم‌های embedded ، نظارت بر سیستم ، اجرای سرویس در سطح سیستم‌عامل باشد. در بعضی از کاربردها استفاده از برنامه‌های شخصی‌سازی شده و یا زبان‌های سطح بالا ممکن نیست و ما کامپایلر نیز در اختیار نداریم. برای همین مفید است که کار با shell در سیستم‌عامل‌های بر پایه یونیکس را بلد باشیم تا بتوانیم وظایف مورد نیاز را در چندین خط اسکریپت بر پایه شل انجام دهیم.

سوال ۱ (۲۵ نمره)

در این بخش می‌خواهیم با استفاده از اسکریپت‌نویسی پروسس‌هایی را که یوزر مشخص کرده و پورتی را اشغال کرده‌اند و پروتکل آن را پیدا کنیم و پورت اشغال شده و پروتکل را نیز نمایش دهیم.
برای این کار نیاز داریم که ابتدا نام پروسس در حال اجرا را از کاربر دریافت کنیم. سپس با استفاده از دستوراتی که برای نشان دادن پورت‌های مورد استفاده در سیستم شما موجود است مانند ss که بخشی از پکیج iproute2 لیست تمامی پروسس‌ها و پورت‌های اشغال شده به همراه پروتکل را بدست آوریم.
به این منظور باید خروجی دستورات مورد نظر را فیلتر کنیم و فقط بخش‌هایی را که مورد نظر ما است پیدا کرده و نمایش دهیم.
خروجی این بخش یک اسکریپت bash است که با گرفتن نام پروسس تمامی پورت‌هایی را که اشغال کرده‌اند را همراه با شماره پورت آن‌ها و پروتکل مورد استفاده نمایش می‌دهد. خروجی باید به صورت سطر و ستون مانند مثال زیر نمایش داده شود.

```
Process Name,    Process Port,    Socket Type
chrome,          5353,            udp
```

مثالی از حالت خروجی سوال ۱

سوال ۲ (۲۵ نمره)

در این بخش می‌خواهیم عملکرد یکی از برنامه‌های معروف سیستم‌عامل‌های بر پایه یونیکس به نام make را شبیه‌سازی کنیم. برنامه make برای ساخت پروژه‌های نرم‌افزاری، فایل‌های سورس و بخش‌های مختلف نرم‌افزار استفاده می‌شود. این برنامه با مقایسه کردن Last Modified Date می‌تواند تغییرات در سورس‌کد را تشخیص داده و فایل‌های پیش‌نیاز برای ساخت برنامه را با دستورهای مشخص شده بسازد. در این بخش ما سعی می‌کنیم با یک اسکریپت بر پایه bash بخشی از قابلیت‌های برنامه make را شبیه‌سازی کنیم.
اسکریپت ما برای ساخت برنامه‌های تحت زبان C/C++ استفاده می‌شود. به این صورت که برنامه ابتدا توسط gcc به فایل آبجکت تبدیل شده و سپس فایل‌های آبجکت مختلف برنامه که توسط gcc تولید شده‌اند با استفاده از linker به فایل اجرایی برنامه تبدیل می‌شوند. به مثال زیر توجه کنید. در این مثال دو فایل main.c و lib.c با gcc کامپایل شده و در آخر با یکدیگر لینک می‌شوند و فایل اجرایی برنامه را می‌سازند.

```
[hossein@TiD bash_test]$ cat main.c
void print_hello();

int main() {
    print_hello();
    return 0;
}
[hossein@TiD bash_test]$ cat lib.c
#include <stdio.h>

void print_hello() {
    printf("hello world!\n");
    return;
}
[hossein@TiD bash_test]$ gcc -c main.c
[hossein@TiD bash_test]$ gcc -c lib.c
[hossein@TiD bash_test]$ gcc -o main lib.o main.o
[hossein@TiD bash_test]$ ./main
hello world!
```

مثالی از دستورات مورد نیاز برای سوال ۲

در این مثال تمامی فایل‌ها با دستور gcc کامپایل شده‌اند. در make این کار به ترتیب انجام می‌شود و فایل‌ها با دستور gcc -c *program.c* کامپایل شده و فایل آبجکت مانند *program.o* تولید می‌کنند. و سپس این خروجی با استفاده از لینکر با کتابخانه‌های زبان C/C++ ترکیب شده و فایل خروجی نهایی ما را می‌سازند
در این بخش ما از دستور gcc برای کامپایل کردن همه بخش‌های نرم‌افزار استفاده کردیم. با استفاده از دستور gcc -v به جای gcc خالی می‌توانید دستوراتی را که این درایور برای کامپایل کردن برنامه شما فراخوانی می‌کند را ببینید که این دستورات پایه کار make هستند که برای راحتی کار ما به استفاده از gcc به صورت تنها بسنده می‌کنیم.
برای حل این تمرین کافی‌است که لیستی از فایل‌های c. و o. بسازید. و تاریخ ادیت شدن را با یکدیگر مقایسه کنید. در صورتی که تاریخ فایل سورس که c. است از فایل o. جدیدتر بود یا اینکه فایل o. وجود نداشت باید برنامه دوباره کامپایل شود. برای راحتی کار می‌توانید تمامی برنامه را به صورت یکپارچه کامپایل کنید.
خروجی این تمرین یک اسکریپت خواهد بود که برای سوال بعدی از آن استفاده خواهید کرد. این اسکریپت باید در صورتی که سورس برنامه ادیت شد، دوباره برنامه را کامپایل کند. در صورتی که برنامه کامپایل نشده بود این اسکریپت باید برنامه را کامپایل کند و در اجراهای بعدی فقط در صورتی که سورس برنامه ادیت شده بود این کار را انجام دهد. همانطوری که اشاره کردیم این کار را با مقایسه تاریخ فایل سورس با فایل آبجکت انجام می‌شود. دستورات ساخت برنامه می‌توانند به صورت مستقیم در این اسکریپت با توجه به مثال بالا داده شود. در نهایت با فراخوانی این اسکریپت در هر بار برنامه ساخته خواهد شد. در صورتی که فایل‌های سورس برنامه دست نخورده بودند برنامه نباید دوباره کامپایل شود.

سوال ۳ (۵۰ نمره)

در این تمرین ما سعی خواهیم کرد که برنامه خودمان را برای شل بنویسیم و از آن برای نوشتن یک اسکریپت ساده استفاده کنیم. برای این کار باید سه برنامه ls ، wc و cat را در محیط لینوکس و با استفاده از زبان C/C++ بنویسیم.
برنامه ls باید لیست فایل‌های موجود در دایرکتوری را به ما بدهد. برنامه cat باید فایل را خوانده و فایل خوانده شده را در stdout بنویسد. برنامه wc تعداد کلمات موجود در فایل ورودی که stdin است را بشمارد و خروجی را به شکل یک عدد به ما تحویل دهد. برای کامپایل کردن این نرم‌افزارها باید از اسکریپت نوشته شده در بخش قبلی استفاده کنید.
خروجی این بخش سه برنامه به زبان C/C++ خواهد بود که عملیات لیست کردن فایل، خواندن فایل و شماردن تعداد کلمات را انجام می‌دهند. دقت کنید که برنامه ls فقط لیست فایل‌ها را به ما می‌دهد که این لیست می‌تواند آدرس relative یا absolute باشد. برنامه cat باید فایل را خوانده و خروجی را در stdout قرار دهد. برنامه wc باید از stdin محتویات فایل را خوانده و تعداد کلمات موجود در آن را برای ما نمایش دهد.

خروجی آخر اسکریپتی خواهد بود که مسیر مورد نظر را از کاربر دریافت کرده و تعداد کلمات فایل‌های درون این مسیر را به ما نشان دهد. طبیعی است که باید از برنامه‌های ls ، cat و wc که خودتان نوشته‌اید استفاده کنید. برای مثال خروجی این بخش به حالت زیر خواهد بود. نیازی به absolute بودن مسیرهای خروجی نیست.

```
file path,                      word count
/home/hossein/bash_test/main.c,    9
/home/hossein/bash_test/lib.c,     9
/home/hossein/bash_test/build.sh,  105
```

مثالی از خروجی اسکریپت سوال ۳

ملاحظات ارسال

- گزارش سوالات باید عملکرد اسکریپت‌ها و برنامه‌های نوشته شده در تمرین را شرح دهد و همچنین تصاویری از عملکرد آن ارائه دهد.
- استفاده از کتابخانه‌های خارجی مانند boost در نوشتن برنامه‌های بالا مجاز نیست. تنها از کتابخانه استاندارد زبان استفاده کنید.
- گزارش باید به صورت lastname_studentnumber و با فرمت pdf باشد.
- فایل گزارش و سورس کدهای نوشته شده باید به فرمت فشرده در سامانه آپلود شود. نام فایل باید به صورت lastname_studentnumber باشد.
- مهلت ارسال تمرین ساعت ۲۳:۵۹ دقیقه روز ۱۸ اسفند می‌باشد
- برای هر تمرین یک هفته مهلت ارسال با تاخیر در نظر گرفته می‌شود که فرمول محاسبه تاخیر به صورت دنباله فیبوناچی خواهد بود و هر تاخیر هر روز با روز قبل جمع خواهد شد. به عنوان مثال دو روز اول ۱ درصد تاخیر و روز دوم ۲ درصد تاخیر و روز سوم ۳ درصد تاخیر و روز چهارم ۵ درصد تاخیر دارد که به همین ترتیب ادامه پیدا می‌کند. در صورتی که چهار روز تاخیر داشته باشید. ۷ درصد جریمه تاخیر لحاظ می‌شود که اگر تا روز آخر که یک هفته است تاخیر وجود داشته باشد ۳۳ درصد جریمه تاخیر لحاظ می‌شود.
- ورژن سیستم‌عامل و توزیع لینوکسی و یا macOS مورد استفاده باید در گزارش درج شود.

موفق باشید.