ปฏิบัติการที่ 6

Shell Interpretive

วัตถุประสงค์

1. สามารถอธิบายการใช้งานและข้อจำกัดของ Command Execution
2. สามารถอธิบายการใช้งานและข้อจำกัดของ Quoting
3. สามารถอธิบายการใช้งานและข้อจำกัดของ Command Substitution
4. สามารถอธิบายการใช้งาน Alias และการปรับตั้งสภาพแวดล้อม

เชลล์รับคำสั่งจากผู้ใช้ได้ 2 ทาง คือ ทาง Command Prompt และทาง Shell Script แม้ว่าคำสั่งที่รับมาตีความจะมีองค์ประกอบไม่ซับซ้อน หรือมีการใช้ Option เพิ่มเติม หรือมีการใช้อักขระพิเศษ โครงสร้างคำสั่งยังคงประกอบด้วย 2 ส่วนเสมอคือ command และ arguments ตัวอย่างเช่น

```
$ ls                                    คำสั่งอาจมีหรือไม่มี Arguments
$ ls lab5
$ find /home -name 'th.po'              Arguments อาจเป็น Path หรือ Pathname
$ find /home -name '*.po' 2>err.log | nl  ใช้การเปลี่ยนทิศทางเพื่อเก็บข้อมูลความผิดพลาด
                                        ไว้ตรวจสอบภายหลัง และส่งผลลัพธ์ผ่าน Pipe
                                        ไปยังคำสั่งต่อไป
```

ตำแหน่งของอักขระพิเศษ (Metacharacters) ในคำสั่งมีความสำคัญต่อการตีความของเชลล์ ผู้ใช้จึงควรศึกษาและทดสอบให้แม่นยำและชัดเจน เพื่อให้ได้ความหมายและผลลัพธ์ที่ถูกต้องตามความต้องการ ในปฏิบัติการนี้จะแนะนำอักขระพิเศษเพิ่มเติม ต่อเนื่องจากปฏิบัติการที่ 5 และแนะนำการใช้งานเชิง Application ต่อไป

ปฏิบัติการ 6.1

Command Execution, Quoting and Command Substitution

ให้ทำปฏิบัติการใน ~/lab6.1/

ไฟล์ข้อมูลเข้า: ไม่มี

Command Execution ประกอบด้วย Sequence และ Group

โดยปกติผู้ใช้ป้อน 1 คำสั่งต่อ 1 Command Prompt ในสถานการณ์ที่ต้องการสั่งงานโดยผู้ใช้ทราบลำดับคำสั่งคือรู้ว่าต้องสั่งคำสั่งอะไรต่อไป เชลล์มีทางเลือกให้ผู้ใช้สามารถป้อนคำสั่งเป็นกลุ่มได้มากกว่า 1 คำสั่งต่อ 1 Command Prompt โดยใช้อักขระ ; แทน Sequence และ อักขระ ( ) แทน Group

1. ให้ทดลองป้อนคำสั่งและสังเกตผลลัพธ์

	ตัวอย่างคำสั่ง	ความหมาย
a)	`pwd > f.log ; ls -Fm >> f.log; date >> f.log`	.....
b)	`who > u.log ; nl < u.log`	.....
c)	`cat > ab ; date >> ab ; nl ab`	.....
d)	`mkdir -p dirA/dirAa; cat > dirA/dirAa/data`	.....
e)	`(cat ; date) > ab ; nl < ab`	.....
f)	`(date ; who | nl ) > nowinfo`	.....

Quoting ประกอบด้วย Back Slash, Single Quote and Double Quote

ผู้ใช้สามารถกำหนดให้เชลล์ตีความอักขระพิเศษต่างๆ เป็นอักขระปกติ (literal value) ด้วยอักขระ Quote ซึ่งมีลำดับความเข้มงวดดังนี้

1) Black Slash จะเข้มงวดที่สุด ไม่มีข้อยกเว้น
2) Single Quote จะยกเว้นกรณี ใช้ Single Quote ซ้อน
3) Double Quote จะยกเว้น $ Back Quote และ Back Slash

หากต้องการให้อักขระเหล่านี้เป็นอักขระปกติให้ Quote ซ้ำ

2. ตัวอย่างต่อไปนี้ จะใช้คำสั่ง echo ตีความ Arguments ให้ทดลองป้อนคำสั่งและสังเกตผลลัพธ์

	ตัวอย่างคำสั่ง	ความหมาย
a)	`echo hello`	*ไม่มีอักขระพิเศษ ไม่จำเป็นต้อง Quote*
b)	`echo "hello world ↵` `are you ready?"`	*ถ้าไม่ Quote อักขระพิเศษถูกตีความ*

c)	`a=5`
	`echo $HOME $a`
	`echo \$HOME`
	`echo '$HOME'`
	`echo "$HOME"`
	`echo 'my home directory is '$HOME''`
		การใช้ Quote กับตัวแปร

d)	`echo "my home directory is pwd"` `echo "my home directory is" pwd`	*ถ้าไม่ Quote พบคำสั่ง ไม่ตีความเป็นคำสั่ง*
e)	`nl /etc/passwd | cut -d':' -f1 | grep \` `'cs56'> student`	*ไม่ตีความ ↵ <enter>*
f)	`find /home -name "*.po" 2>err.log | nl`	*ต้องการให้เป็นชื่อไฟล์หรือกลุ่มของไฟล์*
g)	`who | nl | grep 'cs569'`	*ต้องการให้เป็นข้อความจึงควร Quote ไว้*

Quoting เหมาะสำหรับควบคุมการแสดงข้อความ หรือแสดงข้อความร่วมกับผลลัพธ์ของคำสั่ง มักใช้ในการเขียน Shell Script

Command Substitution ประกอบด้วย `command` หรือ $(command)

หากต้องการนำผลลัพธ์ที่แสดงออกทางจอภาพ (Standard Output) ไปใช้ต่อภายหลัง นอกจากทำโดยการเปลี่ยนทิศทางไปลงไฟล์ (Output Redirection) หรือส่งผลลัพธ์ให้กับคำสั่งอื่นทันทีด้วย Piping (|) แล้ว เชลล์มีวิธีการเปลี่ยนผลลัพธ์จากคำสั่งให้เป็นข้อความเพื่อนำผลลัพธ์นั้นไปประกอบการแสดงผลร่วมกับข้อความอื่นได้ โดยใช้ `command` หรือ $(command)

3. ให้ทดลองป้อนคำสั่งและสังเกตผลลัพธ์

	ตัวอย่างคำสั่ง	**ความหมาย**
a)	`ls hw*`	
	`grep 'the' $(ls hw*)`	
b)	`who`	
	`who | wc -l`	
	`who | wc -l > u.log`	
	`echo "There are $(who | wc -l) users now"`	
	`echo "There are $(who | wc -l) users now" > u.log`	
c)	`cd;pwd`	
	`echo My home directory is cd;pwd`	
	`echo "My home directory is cd;pwd"`	
	`echo "My home directory is $(cd;pwd)"`	

4. จงใช้ Metacharacter ที่เหมาะสมเพื่อให้เชลล์ตีความตามความต้องการดังนี้

	ความต้องการ	**คำสั่งหรือขั้นตอนที่ใช้**
a)	เปลี่ยนไดเรคเทอรี่ชื่อ lab เป็น my lab	
b)	กำหนดตัวแปร a ให้เก็บค่า 44	
	แล้วให้ใช้คำสั่ง echo เพื่อทำการแสดงข้อความ **value of $a is 44** โดยใช้ Single quote	
	ใช้ Backslash	
	ใช้ Double quote	

	ความต้องการ	**คำสั่งหรือขั้นตอนที่ใช้**
c)	สร้างตัวแปร dummy เก็บ path ไดเรคเทอรี่ปัจจุบัน โดยใช้ Command Substitution	

	ใน my lab สร้างไฟล์ dummy จากการ output redirection ดังนี้
	บรรทัดแรก เก็บค่าตัวแปร dummy
	บรรทัดที่สอง เว้นบรรทัด
	บรรทัดที่สาม เก็บ username ของตนเอง (คำสั่ง whoami)
	เช่น
	`$ cat dummy`
	`/home/aws/my lab`
	
	`aws`
		HINT: ใช้ command execution แบบ sequence และ group

คำถาม 6.1

Q1: คำสั่ง `cat > ab ; date>>ab | nl` มีความผิดพลาดอย่างไร? เพราะอะไร?

Q2: คำสั่งใดหมายถึง การดูสิทธิ์ของไฟล์ที่เพิ่งสร้างเสร็จ

a) `touch a | ls -l`
b) `ls -l $(cat > a)`

Q3: จาก ข้อ 4 c) ให้เพิ่มข้อความต่อไปนี้ต่อท้ายไฟล์ dummy

`My user info in /etc/passwd is` <*เลือกเฉพาะข้อมูลของตนเองใน /etc/passwd*>

Q4: สร้างไฟล์เก็บข้อมูลการค้นหาไฟล์ ให้มีข้อมูล 3 ส่วนดังนี้

<*เวลาเริ่มค้นหาไฟล์*>
<*ผลการค้นหา*>
<*เวลาสิ้นสุดการค้นหา*>

Q5: จงเขียนคำสั่งเพื่อแสดงข้อความ

Host info: <*นับจำนวนผู้ใช้ในปัจจุบัน*> users
Time: <*วันเวลาปัจจุบันจากคำสั่ง* `date +%A/%d/%B/%Y%t%r`>
ตัวอย่างเช่น

```
Host info: 6 users
Time: Thursday/09/July/2015      09:08:31 AM
```

Q6: จงจับคู่อักขระพิเศษ (Metacharacter) กับความหมายให้ถูกต้อง

`...`	$(...)	< > >> >&	|	<space>	[...]	*	-	"..."
'...'	~ ./ ../	<enter>	?	$	; (...)	\		

ความหมาย	Metacharacter
1) End of Command (Shell start interpret)	
2) Redirection	
3) Connect the standard output of the left program to the standard input of the right program	
4) Command token separator	
5) Command substitution (back quote or grave accent)	
6) Single quotes	
7) Command substitution	
8) Home directory and Relative Pathname	
9) Matches one character given in the bracket	
10) Double quotes	
11) Matches any number of any characters including none	
12) Matches any single character	
13) Escape next character	
14) Command execution	
15) Variables reference	
16) Option of command	

ปฏิบัติการ 6.2

Useful Command: tee xargs and tar

ให้ทำปฏิบัติการใน ~/lab6.2/

ไฟล์ข้อมูลเข้า: ไม่มี

คำสั่ง tee ทำหน้าที่คล้ายคำสั่ง cat โดย tee มี Default Standard Stream ครบทั้ง 3 Stream ทำหน้าที่รับข้อมูลจาก Standard Input (คีย์บอร์ด) มาแสดงออก 2 ช่องทาง คือ Standard Output (จอภาพ) และเก็บผลลัพธ์เดียวกันนั้นลงไฟล์ซึ่งระบุไฟล์เป็น Arguments ได้มากกว่า 1 ไฟล์

5. ให้พิจารณาและทดลองป้อนคำสั่งต่อไปนี้

	ตัวอย่างคำสั่ง	ความหมาย
a)	`tee txt`	*รับข้อมูลจาก Stdin, แสดงทาง Stdout, และเก็บลงไฟล์ txt*
b)	`tee txt n3 n4 n5`	
c)	`tee < txt n3 n4 n5`	
d)	`tee < txt n3 n4 > n5`	
e)	`who | tee txt n3 n4 n5`	

6. หากแบ่งส่วนคำสั่งด้วย | ให้ป้อนคำสั่งทีละส่วน โดยใช้คีย์ลูกศรขึ้นลง (History) เรียกคืนคำสั่งเดิม

	คำสั่ง	ความหมาย
a)	`who`	
	`who | nl`	
	`who | nl | grep 'cs54'`	
	`who | nl | grep 'cs54' | tee wList`	
b)	`nl`	
	`nl | tee data`	
	`nl | tee data | wc -l`	
	`nl | tee data /dev/tty`	
	`nl | tee data /dev/tty > outf`	
c)	`find /home -name "*.po"`	
	`find /home -name "*.po"  2>/dev/null`	
	`find /home -name "*.po"  2>/dev/null | tee txt`	
	`find /home -name "*.po"  2>/dev/null | tee /dev/tty | nl`	

คำสั่ง xargs ทำหน้าที่แบ่งข้อความที่รับจาก Standard Input ไปเป็น รายการ Arguments ให้กับคำสั่งตามจำนวนที่ต้องการ จากคำสั่ง man จงบอกความหมายของ Option ต่อไปนี้

Option	ความหมาย
-n	
-p	
-t	
--show-limits	

7. ตัวอย่างการแบ่งขนาด arguments ของ xargs ให้พิจารณาและทดลองป้อนคำสั่งต่อไปนี้

	ตัวอย่างคำสั่ง	ความหมาย
a)	`echo a b c d e f g`	
b)	`echo a b c d e f g | xargs`	
c)	`echo a b c d e f g | xargs -n2`	*แบ่งข้อความ*
d)	`ls | nl`	*ตรวจสอบจำนวนไฟล์ในไดเรคเทอรี่ปัจจุบัน*
e)	`ls | xargs`	
f)	`ls | xargs | nl`	
g)	`ls | xargs -tn2`	*-t จะแสดงคำสั่งที่รับ args ไปทำงานต่อ* *คำสั่ง echo จะถูกเรียกใช้ หากไม่ระบุคำสั่งมารับ*
h)	`ls | xargs -n2 | nl`	
i)	`ls *.dat | xargs -tn2 wc -l`	*คำสั่ง wc -l จะรับข้อความไปเป็น arg ครั้งละ 2 ข้อความ*

การกำหนด Arguments ให้กับคำสั่ง นอกจากกำหนดให้โดยตรงยังทำได้โดยวิธีอื่นดังนี้

`$ grep 'the' f1 f2 f3`	*รับผ่าน Arguments โดยตรง*
`$ cat f1 f2 f3 | grep 'the'`	*รับผ่าน Piping*
`$ grep 'the' $(ls f*)`	*รับผ่าน Command Substitution*
`$ ls f* | xargs -n 2 grep 'the'`	*รับผ่าน xargs จะสามารถกำหนดขนาดที่ต้องการได้*

8. คำสั่ง xargs จะแบ่ง Arguments เป็นส่วนๆ ส่งให้กับคำสั่งที่ไม่สามารถทำงานกับ Arguments จำนวนมากได้ ให้ทดลองและสังเกตผลลัพธ์จากการใช้คำสั่ง xargs กับคำสั่ง find และเนื่องจากเป็นการลบไฟล์ ดังนั้นให้สำเนา ~dummy/lab6/mediawiki-1.26.2 มาเป็น media1, media2, media3,media4, media5 และ media6 ใน ~/lab6.2 ตามลำดับ เพื่อเตรียมไว้ใช้ภายหลัง

	ตัวอย่างคำสั่ง	ความหมาย
a)	`find media1 -name "*.gif" | nl`	*มีทั้งหมด 41 ไฟล์*
b)	`find media1 -name "*.gif" | xargs -n10 | nl`	
c)	`find media1 -name "*.gif" -exec rm -f {} \;`	*100 files -> rm has to executed 100 times.* *rm –f file1;rm –f file 2; rm –f file3;...*
d)	`find media2 -name "*.gif" | xargs rm -f`	*ทำครั้งเดียว* *rm –f <all found files>*
e)	`find media3 -name "*.gif" | xargs -tn20 rm -f`	*แบ่งกลุ่มทำ* *100 files -> rm will invokes 5 times* *rm –f file1 file2 file3 file4*

f)	`time find media4 -name "*.gif" -exec rm -f {} \;`
	`time find media5 -name "*.gif" | xargs -t rm -f`
	`time find media6 -name "*.gif" | xargs -tn20 rm -f`
		ลองเปรียบเทียบความเร็วโดยใช้คำสั่ง time

คำสั่ง tar (Tape Archive) ทำหน้าที่สร้างหน่วยข้อมูลของกลุ่มไฟล์ที่ไม่ต้องการเปลี่ยนแปลง แก้ไขอีกต่อไป (Archive) จากคำสั่ง man จงบอกความหมายของ Option ต่อไปนี้

Option	ความหมาย
-c	
-f	
-t	
-v	
-x	
-z	

9. ให้พิจารณาและทดลองป้อนคำสั่งต่อไปนี้ (ให้ทำปฏิบัติการใน ~/lab6.2/)

	ตัวอย่างคำสั่ง	ความหมาย
a)	`tar -cf po.tar ../lab2/*.po`	*สร้าง Archive จากกลุ่มไฟล์*
	`file lab2.tar`	*และดูชนิดของ Archive ที่ได้*
b)	`tar -cvf lab2.tar ../lab2`	*View ขณะสร้าง Archive*
c)	`tar -tf lab2.tar`	
d)	`tar -tvf lab2.tar`	
e)	`tar -xvf lab2.tar`	*Restore ไฟล์จาก Archive*
f)	`tar -cvzf lab2.tar.gz ../lab2`	*หากสร้าง Archive พร้อมบีบอัดข้อมูล (z or Z)*
		ต้องระบุ z or Z เมื่อต้องการ view หรือ restore

10. แม้ว่าการสร้างกลุ่มไฟล์ด้วยคำสั่ง tar จะมีความหมายต่างจากการสำรองข้อมูล (Backup) แต่ผลลัพธ์จากคำสั่ง tar สามารถนำมาปรับใช้กับงานสำรองข้อมูลได้ ซึ่งผู้ใช้จะเรียกข้อมูลที่สำรองไว้กลับคืน (Restore/Extract) เมื่อมีปัญหาเกิดขึ้นกับข้อมูลจริงที่กำลังถูกใช้งาน ให้เปรียบเทียบผลลัพธ์จากการสำรองข้อมูล ~/lab2 ด้วยคำสั่ง cp zip และ tar ดังนี้

	คำสั่ง	ขั้นตอน
a)	`cp -r ~/lab2 lab6case1`	*เตรียมข้อมูลสำหรับทดสอบทั้ง 3 คำสั่ง*
	`cp -r ~/lab2 lab6case2`	
	`cp -r ~/lab2 lab6case3`	
b)	`cp -r lab6case1 backupdir`	*ทดลองสำรองข้อมูลกรณีที่ 1*
		ดู Attributes ของ backupdir
		ดู Attributes ของ content ของ lab6case1
		ดู Attributes ของ content ของ backupdir
		Attributes เหมือนกันหรือไม่?
c)	`zip -r backup.zip lab6case2`	*ทดลองสำรองข้อมูลกรณีที่ 2*
		ดู Attributes ของ backup.zip
	`less backup.zip or`	*ดู Attributes ของ content ของ backup.zip*
	`unzip -l backup.zip`	*ดู Attributes ของ content ของ lab6case2*
	`unzip backup.zip`	***Attributes เหมือนกันหรือไม่?***

	คำสั่ง	ขั้นตอน
d)	`tar -cvf backup.tar lab6case3`	*ทดลองสำรองข้อมูลกรณีที่ 3*
		ดู Attributes ของ backup.tar
	`tar -tvf backup.tar`	*ดู Attributes ของ content ของ backup.tar*
		ดู Attributes ของ content ของ lab6case3
		Attributes เหมือนกันหรือไม่?
	`tar -xvf backup.tar`	

ในการสำรองข้อมูล สิ่งสำคัญคือเมื่อเรียกข้อมูลกลับคืน คุณลักษณะของไฟล์ (File Attributes) เช่น สิทธิ์ เจ้าของ วันเวลาแก้ไข ต้องคงเดิม ณ วันเวลาที่สำรองข้อมูล จากผลลัพธ์จากคำสั่งข้างต้น กรณีใดเหมาะกับการสำรองข้อมูลมากที่สุด เพราะอะไร?

คำถาม 6.2

Q7: `who | tee w2` เทียบได้กับ คำสั่งใด?

a) `who > temp ; tee < temp w2`
b) `who > temp ; tee < temp > w2`
c) `who > temp ; tee temp w2`
d) `who > temp ; tee temp > w2`

Q8: จงค้นหาไฟล์ *.xml จาก ~/lab6.2/media1 เพื่อสำเนามาไว้ที่ ~/bak โดยใช้ xargs แบ่งทำสำเนาครั้งละ 3 ไฟล์

Q9: จงอธิบายผลลัพธ์ของคำสั่ง

```
ls *.dat | xargs -t -I {} mv {} {}.bak
```

Q10: จงสร้าง Archive ชื่อ xml.tar ของ ไฟล์ *.xml จาก ~/lab6.2/media1 -r

Q11: จากปฏิบัติการ คำสั่ง tar หากต้องการเรียกคืน Archive บางส่วนคือเฉพาะ ~/lab2/lang ต้องใช้คำสั่งอย่างไร?

Q12: จงสร้าง Archive และแตก Archive เพื่อเปรียบเทียบความแตกต่างระหว่าง

a) `cd ~/lab2; tar -cvf lab2.tar .`
b) `cd; tar -cvf lab2.tar lab2`
c) `cd ~/lab6.2; tar -cvf lab2.tar ~/lab2`

Q13: การสร้าง Archive ของ ~dummy/lab2 โดยใช้ Absolute Pathname ต่างจากการใช้ Relative Pathname อย่างไร?

Q14: การสร้าง Archive ควรกำหนดไดเรคเทอรี่เป้าหมาย ด้วย Relative Pathname หรือ Absolute Pathname เพราะเหตุใด?

Q15: ถ้าผู้สอนไม่ต้องการให้น.ศ. นำ lab-solution.tar.gz ไปใช้ ผู้สอนต้องทำอย่างไรบ้าง?

ปฏิบัติการ 6.3

Alias and Shell Customization

ให้ทำปฏิบัติการใน ~

ไฟล์ข้อมูลเข้า: ไม่มี

Alias คือ การกำหนดชื่อหรือข้อความให้ทำหน้าที่เป็นคำสั่ง รูปแบบการตั้ง Alias เป็นดังนี้

```
$ alias name='command with options'
```

เมื่อป้อนชื่อ Alias ที่ Prompt เชลล์จะตีความตามข้อความที่ตั้งไว้ จากปฏิบัติการ 5.2 ชื่อที่ถูกตั้งจาก alias จะมีความสำคัญสูงที่สุดในการตีความคำสั่งของเชลล์ ตัวอย่างการตั้ง Alias เช่น

a) `alias ll='ls -l'`
b) `alias clear='clear;ls'`
c) `alias ..='cd ..'`
d) `alias x='exit'`

11. จากตัวอย่างการใช้คำสั่ง alias ต่อไปนี้ จงบอกความหมาย

	ตัวอย่างคำสั่ง	ความหมาย
a)	`alias`	*ดูรายการ Alias ที่ตั้งไว้*
b)	`alias ll`	
	`ll -t`	
c)	`\clear`	
d)	`unalias ll`	

การปรับตั้งค่าสภาพแวดล้อม (Shell Customization) คือ การตั้งค่าการทำงานของเชลล์ เพื่อให้ได้ข้อมูลที่จำเป็นต้องรู้ ที่ต้องใช้งานบ่อยๆ เช่น Working Directory หรือตั้ง Alias ให้กับคำสั่งที่ใช้บ่อยๆ เป็นต้น

วัตถุประสงค์เพื่อลดขั้นตอน ให้ใช้งานผ่าน Command Line ได้สะดวก โดยจะเป็นการตั้งค่าตัวแปรระบบ (System Variables) การปรับตั้งค่าทำได้ 2 วิธี

วิธีที่ 1 ทำโดยตรงที่ Command Line วิธีนี้เป็นการปรับแต่งที่ให้ผลแบบชั่วคราว

วิธีที่ 2 ทำที่ไฟล์ตั้งต้นของเชลล์ (Individual user configuration files) โดยไฟล์ตั้งต้นจะถูกอ่านทุกครั้งที่ผู้ใช้ Login เมื่อปรับแต่งแล้วให้ใช้คำสั่ง source ตามด้วย Configuration file ที่แก้ไข

12. โดยปกติเมื่อผู้ใช้ Login สำเร็จ เชลล์เริ่มต้นตามที่ตั้งไว้ใน /etc/passwd จะทำงาน การตั้งค่าสภาพแวดล้อมจะเริ่มต้นจาก /etc/profile และ ~/.bash_profile หรือ ~/.bash_login หรือ ~/.profile ถ้ามีไฟล์ใดไฟล์หนึ่งตามลำดับ เมื่อผู้ใช้ Logout เชลล์จะอ่าน ~/.bash_logout ถ้ามี

    12.1 จงสำรวจใน ~ ว่ามี Configuration files ใดบ้าง

    12.2 จาก 12.1 ถ้าไม่มีไฟล์ใด ให้สร้างขึ้นเองและเพิ่มคำสั่งแสดงข้อความว่า “This is <*Conf. files*>”

    12.3 ทดลอง Login เพื่อสังเกตลำดับการเรียก Configuration files

13. จงปรับตั้งสภาพแวดล้อมของการทำงานกับเชลล์ ที่ ~/.bash_profile ดังนี้

    13.1 จากคำสั่ง man bash ได้แนะนำวิธีเปลี่ยนรูปแบบ Primary Prompt $PS1 ให้ค้นหาคำ PS1 และ PROMPTING เพื่อศึกษาการตั้งค่า $PS1 ให้มีรูปแบบดังนี้

    [เวลา] Useraname : ลำดับที่ของคำสั่ง [ไดเรคเทอรี่ปัจจุบัน]

    ตัวอย่างเช่น

    ```
    [12:12:10] aws:18 [~] $
    ```

    13.2 ตั้ง alias ดังนี้

Alias name	ความต้องการ
a) `..`	เปลี่ยนไดเรคเทอรี่ ขึ้นไปที่ Parent ของ ไดเรคเทอรี่ปัจจุบัน
b) `ls`	ls –F เพื่อแสดงสัญลักษณ์แทนประเภทไฟล์ต่างๆ
c) `ls -l`	ls -lF
d) `find <filename>`	ใช้ค้นหา <filename> ใน ~ แสดง Pathname และ Attributes เมื่อค้นพบ
e) `cal`	แสดงปฏิทินเดือนปัจจุบันและเดือนต่อไป 1 เดือน
f)* `path`	`"echo -e '${PATH//:/\\n}'|nl"`

**ที่มา: www.askapache.com/linux/bash-power-prompt.html*

คำถาม 6.3

Q16: จากการตั้ง alias หากต้องการใช้คำสั่ง cal ให้แสดงปฏิทินเดือนปัจจุบันตามปกติ ไม่ใช่ alias ชื่อ cal โดยไม่ unalias ต้องทำอย่างไร?

Q17: ต้องการยกเลิกชื่อทั้งหมดที่ตั้งจากคำสั่ง alias ต้องทำอย่างไร?

Q18: เมื่อ $PS1 คือ Primary Prompt แล้ว $PS2 $PS3 $PS4 เป็น Prompt สำหรับสถานการณ์ใด?

Q19: ให้ปรับแต่งสภาพแวดล้อม

a) แสดงเดือนและเน้นวันที่ปัจจุบัน เมื่อ log in
b) exit และ logout เป็นคำสั่งเพื่อออกจากโฮสต์ จงตั้ง Alias ชื่อ exit และ logout ให้แทนการสรุปจำนวนไฟล์ที่เพิ่มขึ้นจากการเข้าใช้ครั้งก่อน และให้ exit หรือ logout อีกครั้ง

ปฏิบัติการ 6.4
แบบฝึกหัดเพิ่มเติม

ให้ทำปฏิบัติการใน ~/lab6.4

ไฟล์ข้อมูลเข้า: ~dummy/lab6/mediawiki-1.26.2/

---

14. จงค้นหาไฟล์ที่มีตัวเลขเป็นส่วนประกอบภายใน ~dummy/lab6/mediawiki-1.26.2/ ด้วยวิธีการต่อไปนี้
    a) ให้เปลี่ยนทิศทางของ Standard output ไปลง ~/lab6.4/out.log และ Standard error ไปลง ~/lab6.4/err.log
    b) ให้เปลี่ยนทิศทางของ Standard output และ Standard error ไปลงที่ไฟล์เดียวกันที่ ~/lab6.4/found.log
    c) ให้แสดงลำดับที่ และเก็บผลลัพธ์ลงไฟล์ ~/lab6.4/num-found.log
    d) หากเริ่มค้นหาตั้งแต่ /home จะพบไฟล์และความผิดพลาดจำนวนมาก ให้สนใจเฉพาะ Output ที่เป็นที่อยู่ของไฟล์ที่พบ และไม่สนใจความผิดพลาดทั้งหมด หากใช้ Error Redirection จะได้ไฟล์ขนาดใหญ่ ดังนั้นจะทำอย่างไรเพื่อควบคุมความผิดพลาดที่พบ ไม่ให้แสดงทางจอภาพ
    e) จาก d) ให้แสดงลำดับที่ และจำนวนไฟล์ที่พบ แสดงทาง Standard Output และเก็บลงไฟล์ ~/lab6.4/found.log
    f) ให้สำรองข้อมูลที่พบด้วยคำสั่ง tar และไม่แสดงความผิดพลาดทางจอภาพ
    g) ให้สำเนาไฟล์ที่พบ ไปไว้ใน ~/lab6.4/log ใช้ –exec และไม่แสดงความผิดพลาดทางจอภาพ
    h) ให้สำเนาไฟล์ที่พบ ไปไว้ใน ~/lab6.4/log ใช้ xargs และไม่แสดงความผิดพลาดทางจอภาพ

15. กำหนดคำสั่งเริ่มต้นให้ดังนี้

```
last|grep $(whoami)|cat -n
```

15.1 จงทำคำสั่งให้สมบูรณ์ เพื่อให้ได้ผลลัพธ์ต่อไปนี้

*(***ตัวเลขที่แสดงขึ้นกับจำนวนครั้งที่เข้าใช้โฮสต์ และชื่อผู้ใช้ขึ้นกับผู้เรียกคำสั่ง)*

```
$ last|grep $(whoami)|cat -n ...
     1  aws      pts/2        ppp-202-176-128- Mon Feb 15 04:39   still logged in
     2  aws      pts/0        ppp-202-176-128- Mon Feb 15 04:29 - 04:29  (00:00)
     3  aws      pts/0        ppp-202-176-128- Mon Feb 15 04:27 - 04:27  (00:00)
     4  aws      pts/0        ppp-202-176-128- Mon Feb 15 04:27 - 04:27  (00:00)
     5  aws      pts/0        ppp-202-176-128- Mon Feb 15 04:25 - 04:26  (00:00)
     6  aws      pts/0        ppp-202-176-128- Mon Feb 15 04:21 - 04:25  (00:03)
     7  aws      pts/0        202.14.164.246   Fri Feb 12 14:42 - 14:45  (00:02)
aws was login 7 times
```

15.2 ทำอย่างไร? เมื่อต้องการเก็บผลลัพธ์แบบเดียวกันลงไฟล์ num-login.log

16. พิจารณาการทำงานของคู่คำสั่งต่อไปนี้ โดยให้อธิบายด้วยหลักการตีความของเชลล์ และให้สังเกตว่าผลลัพธ์เหมือนกันหรือไม่?

	i	ii	ความหมาย
a)	`echo aws | last`	`last $(echo aws)`	*แสดงประวัติการเข้าใช้โฮสต์ของ aws*
b)	`tee < flist n3 n4 > n5`	`tee flist n3 n4 n5`	.....
c)	`mkdir `cat dirList``	`mkdir < dirList`	.....
d)	`who | nl`	`nl $(who)`	.....
e)	`grep 'cs' | nl`	`nl < grep 'cs'`	.....
f)	`cat fileB | nl`	`nl < fileB`	.....

17. จงใช้ Redirection Piping Quoting Command Executions Command Substitutions เพื่อทำงานดังนี้

a) สร้างไฟล์ที่มีข้อมูลของไฟล์ต่างๆ ต่อกัน

b) นับไฟล์ที่ลงท้ายด้วย .c

c) สร้างไดเรคเทอรี่ชื่อเดียวกับชื่อผู้ใช้

d) ดู Attribute ของไดเรคเทอรี่ปัจจุบัน

e) ดูคุณลักษณะของไฟล์ตามชื่อไฟล์ใน flist

f) สร้างไดเรคเทอรี่ตามชื่อไฟล์ที่เป็นข้อมูลของ filelist

g) นับเฉพาะไดเรคเทอรี่ที่อยู่ใน ~

h) ค้นหาไฟล์ i18n เปลี่ยนชื่อเป็น I18N และนับว่าทำไปกี่ไฟล์

i) แสดงเฉพาะไดเรคเทอรี่ในไดเรคเทอรี่ปัจจุบัน

ตัวอย่าง 17.1 "นับจำนวนผู้ใช้ในขณะนี้"

```
จำนวนผู้ใช้ทั้งหมด 3 คน
ณ เวลา ....
1 cs53996 …
2 cs53564 …
3 cs52111 …
```

ขั้นตอนที่ 1 ใช้คำสั่งใดบ้าง?

คำสั่งที่ใช้ คือ

a. who แสดงผู้ใช้ปัจจุบัน
b. wc –l นับจำนวนบรรทัด หรือ nl ใส่เลขบรรทัด
c. เลือกแสดงเฉพาะเวลา จากคำสั่ง date โดยใช้ option…

ขั้นตอนที่ 2 เชื่อมด้วยอะไร? > < | ; (…) หรือ $(…)

ตัวอย่าง 17.2 จาก โจทย์ ข้อ d) ดู Attribute ของไดเรคเทอรี่ปัจจุบัน

ขั้นตอนที่ 1 ใช้คำสั่งใดบ้าง?

คำสั่งที่ใช้คือ

a. ls –ld ทำหน้าที่...
b. pwd ทำหน้าที่...

ขั้นตอนที่ 2 เชื่อมด้วยอะไร?

```
ls -ld | pwd
ls -ld > pwd
ls -ld < pwd
ls -ld $(pwd)
ls -ld ; pwd
(ls-ld; pwd)
pwd < ls -ld
pwd | ls -ld
```