시작하기 전: 사수에게 unix-lecture.tar.gz을 풀어달라고 하고 시작할 것!!!

Course 1: 아주 기본

1. exit

logout하는 command는 아주 간단하다. exit 을 커맨드에서 입력하고 return 키를 누르면 logout 된다.

2. ls & command option

현재 directory에서 file의 list를 보는 명령어, 즉 DOS에서의 dir에 해당하는 명령어로는 'ls'가 있다. (list의 약어.) ls 커맨드를 실행시키면 다음과 같이 표시된다.

```
[user@mlet1 ~]$ ls
hello.c   practice/    sincos.x*  sine.x*
hello.x*  sincos1.x*  sine.c     unix-lecture/
```

많은 유닉스 command에는 수많은 option들이 있는데 이는 ls도 마찬가지이다. 이 중 가장 많이 사용하는 option 두 가지는 a(all)와 l(long)이 있다. (unix manual 참조) command에 option을 적용시키기 위해선 - 를 붙이면 된다.

a option을 적용시켰을 경우 다음과 같은 출력 결과를 얻을 수 있다.

```
[user@mlet1 ~]$ ls -a
./           .bashrc  hello.c   sincos1.x*  .ssh/         .Xauthority
../          .canna   hello.x*  sincos.x*   .tcshrc       .xemacs/
.bash_logout   .emacs   .kde/     sine.c     unix-lecture/  .zshrc
.bash_profile  .gtkrc   practice  sine.x*     .viminfo
```

즉, UNIX에서 파일 명 앞의 방점으로 표시되는 hidden files를 표시해주는 option이 a이다. 또한 l option을 적용시켰을 경우 출력 결과는 다음과 같다.

```
[user@mlet1 ~]$ ls -l
total 80
-rw-r--r-- 1 user1 users    85 2010-07-14 18:02 hello.c
-rwxr-xr-x 1 user1 users 12918 2010-07-14 18:02 hello.x
drwxr-xr-x 2 user1 users  4096 2010-07-14 18:03 practice
-rwxr-xr-x 1 user1 users 12919 2010-07-14 18:05 sincos1.x
-rwxr-xr-x 1 user1 users 12919 2010-07-14 18:05 sincos.x
-rw-r--r-- 1 user1 users   144 2010-07-14 18:04 sine.c
-rwxr-xr-x 1 user1 users 12919 2010-07-14 18:05 sine.x
drwxr-xr-x 2 user1 users  4096 2010-07-14 18:05 unix-lecture
```

여기서 앞의 drwxr-xr-x와 같은 표시는 directory 여부 (d,-), permission을 나타내며, 기타 DOS에서

역시 볼 수 있었던 byte 수, 수정 시간 등을 나타내 줌을 볼 수 있다. 두 옵션을 모두 적용하기 위해선
[user@mlet1 ~]$ ls -al
와 같이 입력하면 된다.

3. file 다루기

Windows에서 흔히 file을 다룰 때 이름 바꾸기와 삭제, 잘라내기, 복사, 붙여넣기 등을 행한다. UNIX에서도 (같은 '파일'을 다루는 것이므로) 비슷한 일들을 행하는데, 이를 위한 커맨드는 5가지가 아닌 3가지이다. 이는 잘라내기와 이름 바꾸기가 통합되어 있고, Windows에서의 '클립보드'를 사용하지 않기 때문이다. 이 세 가지 커맨드를 알아보면

1) cp : copy

cp [object] [target] 과 같이 쓴다. 예를 들어,

```
[user@mlet1 ~]$ cp sine.c sine1.c
```

와 같이 쓰면 sine.c와 같은 내용의 sine1.c 를 복사시킨다.

2) rm : remove

rm [object] 와 같이 쓴다. 예를 들면,

```
[user@mlet1 ~]$ rm sine1.c
```

와 같이 쓰면 sine1.c를 지운다. (alias 설정에 따라 지울 것인지 묻는 경우도 있다.) -i, -f 옵션이 있는데, -i는 지울 것인지 묻고 지우는 일을, -f는 강제적으로 지우는 일을 행한다.

3) mv : move, rename

mv는 '파일을 이동하는' 명령어이다. 즉, cut&paste , 그리고 Windows에서의 'rename' 명령을 모두 행할 수 있다. 이의 용법은 cp와 같다. 즉,

```
[user@mlet1 ~]$ mv sine.c sine1.c
```

라고 하면 sine.c는 sine1.c 로 바뀐다.

4. directory

현재 디렉토리를 볼 수 있는 커맨드는 pwd (print working directory) 이다. 예를 들면,

```
[user@mlet1 ~]$ pwd
/home/user
```

와 같이 표시된다. / directory는 root directory로, 그 곳에서 모든 경로들이 시작된다(Windows의 '내 컴퓨터'와 비슷). DOS와 같이, cd (change directory) 명령어를 이용하면 directory를 바꿀 수 있다. unix에서는 ./ 는 현재 디렉토리, ../ 는 상위 디렉토리를 뜻한다.

디렉토리를 만들기 위해서는 mkdir, 지우기 위해서는 rmdir 명령어를 이용한다. 디렉토리 안의 모든 내용이 지워지지 않았을 때에 rmdir 명령어는 그것을 행하지 않으며, 이 때에는 rm 명령어를 사용하여야 한다. 디렉토리 안의 모든 내용과 함께 directory를 삭제할 경우, -r (recursive) 옵션을 이용한다. 예를 들어

```
[user@mlet1 ~]$ rm -rf unix-lecture/
```

라고 입력하면 unix-lecture/ 안의 모든 파일들을 지운다.

cp, mv 등의 모든 명령어에 directory를 넣어 행할 수 있다.

예를 들어

```
[user@mlet1 ~]$ mkdir newdir
[user@mlet1 ~]$ cd newdir
[jwpk1201@mlet1 ~/newdir]$ cp ../sine.c sine.c
```

와 같이 입력하면 상위 디렉토리의 sine.c가 하위 디렉토리에 복사된다.

Course 2: 조금 덜 기본

1. logout, password
1) logout : exit, logout
2) password 변경 : passwd
또한 보안에 아주 중요한 'password 변경'은 UNIX 안에서 행하는데, 이는 passwd 커맨드를 실행시키면 된다. 종전 비밀번호와 비슷하거나, 숫자로만 되거나 알파벳으로만 된 비밀번호는 시스템이 거부할 가능성이 크다.

2. file list

ls : 숨김파일을 제외한 파일 목록을 보여줌
ls -a : 숨김파일을 포함한 전부를 보여줌
ls -l : 파일의 자세한 정보를 보여줌
ls -al과 같이 쓰면 a와 l 두 옵션을 모두 적용시킨다.
ls -t 시간순 정렬

4. directory 다루기: 우선 directory의 개념을 익힐 것!
모든 디렉토리는 "/" 부터 출발한다. 가령, /home/user1/unix-lecture처럼. 맨 앞의 "/"을 빼면 현재에 있는 디렉토리에서부터 출발한 위치를 가르친다.
pwd : 현재 경로를 보여줌
mkdir xxx: xxx 디렉토리를 만듬
rmdir xxx: xxx 디렉토리를 지움 (비어있을 때만 성공)
cd xxx: xxx디렉토리로 옮겨감. xxx를 주지 않고 cd만 하면 자신의 홈디렉토리로 옮겨감

```
cd /tmp
pwd
ls
cd /usr/local
pwd
ls -l
cd
pwd
ls -l
cd /home/user1/unix-lecture
pwd
cd ..
pwd
cd unix-lecture
pwd
cd ..
cd ./unix-lecture
pwd
```

```
cd ../practice
pwd
cd
```

3. file 다루기

copy : cp [object] [target]
move : mv [object] [target]
remove : rm [filename]
rm -r : 디렉토리와 디렉토리 안의 모든 것을 지움
rm -f : 지울 때 아무것도 묻지 않고 지움 (default로 안 물어보고 지움: 지울 때 주의!!!)
rm -i : 지울 때 지울 여부를 묻고 지움

cp A B copy, A를 B로 복사. A or B에 경로를 지정해 줄 수 있다.
  -r 디렉토리 복사
  ex) cp /home/user_1/linux ./unix/.
  ㄴ이 때 첫 번째 .은 현재 경로를, 두 번째 .은 같은 파일 이름을 뜻한다.

rm A remove, 파일 지우기.
  -i interaction, 지울지 말지 물어보기
  -f 묻지 않고 지우기
  -r 디렉토리 지우기
  -R 디렉토리 지우기
  ex) rm -rf ./linux/unix

mv A B move, A를 B로 옮기기 / A의 이름을 B로 바꾸기
  리눅스에서 mv는 먼저 파일A를 지운 다음 B로 재생성하는 명령어이다.
  경로상에 같은 이름이 존재하면 덮어씌워버리게 된다.

cat A A의 내용을 화면에 출력한다.

```
cat hello.c
```

grep A B B에 해당하는 파일 내부에서 A가 사용된 line을 보여준다.
  ex) grep printf *std
  ㄴ이 때 *은 모든 것을 포함하는 문자이며,
  이 경우 현재 경로 내에서 이름이 std 로 끝나는 모든 파일을 지정한다.

```
grep Hello hello.c
grep hello hello.c
grep -i hello hello.c
grep -i heLLO hello.c
grep -v Hello hello.c
grep -E 'Hello|stdio' hello.c
grep stdio *.c
```

head A　　A의 첫 10줄을 화면에 출력한다.
　　-n19　첫 19줄을 출력(임의의 숫자 가능)

```
head -n 3 hello.c
head -3 hello.c
```

tail A　　A의 마지막 10줄을 화면에 출력한다.
　　-n19　마지막 19줄을 출력(임의의 숫자 가능)

```
tail -n 3 hello.c
tail -3 hello.c
```

cut -c [num] A 어떤 조건을 바탕으로 각 줄의 num번째 만큼 화면에 출력한다.
cut -f [num] A 어떤 조건을 바탕으로 각 줄의 num번째 만큼 화면에 출력한다.
　　-c [num] character, 글자의 개수를 기준
　　-f [num] -d '□' A
　　field, □를 기준으로 하는 블록에서 num의 조건에 따라 출력
　　ex) cut -f 1 -d ':' A
　　└':'을 기준으로 하는 field의 첫번째를 출력
　　└num : N　N번째 필드만 출력(구분 문자 비포함)
　　N-　N번째 이후 필드 모두 출력(첫 구분 문자 비포함)
　　-M　M번째 까지의 필드 모두 출력(마지막 구분 문자 비포함)
　　N-M　N번째 부터 M번째 까지의 필드 모두 출력
　　(첫 번째와 마지막 구분 문자 비포함)

```
cut -c 3-5 hello.c
cut -c -2 hello.c
cut -f 2 -d ' ' hello.c
```

wc -□ A　　word count, 단어 수 세기
　　-l　line 세기
　　-w　단어 수 세기(space로 구분된 것을 하나의 단어로 간주함)
　　-c　문자 수 세기

```
wc -l hello.c
```

[CMD] > A　어떠한 comand에 의해 생긴 내용을 A에 덮어쓴다.
[CMD] >> A　어떠한 comand에 의해 생긴 내용을 A에 이어쓴다.

```
cut -f 2 -d ' ' hello.c > test.file
cat test.file
ls -l >> test.file
cat test.file
rm test.file
```

sort [inputs]　input들을 알파벳 순으로 정렬한다.

```
sort hello.c
```

uniq　　연달아서 나오는 내용들을 축약하여 한번만 화면에 표시한다.

-c 몇번 연달아서 나왔는지 숫자로 표시해 준다.

```
uniq hello.c
uniq -c hello.c
```

[CMD1] | [CMD2] 어떠한 cmd1에 의해 생긴 내용을 cmd2 의 stdin으로 사용한다.

ex1) head -n 20 unix | tail -n 3

└unix라는 파일의 18~20번째 줄 출력

ex2) grep printf *.c | cut -f 1 -d':' | sort | uniq -c

└.c 로 끝나는 파일에 printf가 각각 몇번 포함되어있는지 정렬

```
uniq -c hello.c | sort
```

find [dir] -name A

dir라는 이름의 directory아래에서 A라는 이름을 가진 파일을 검색

```
[example]
    find / -name hello.c
```

man [cmd] 어떠한 cmd의 manual. [cmd] --help역시 같은 기능이다.

diff 파일의 서로 다른 부분을 확인

-b 띄어쓰기 무시하고 비교

-q 다른 파일의 경우 다르단 메시지만 표시

```
[example]
    diff sine.c hello.c
    diff -q sine.c hello.c
```

clear 화면 초기화

top 현재 컴퓨터 cpu memory 사용량 표시

1 누르면 컴퓨터 내 개개의 cpu의 동작 확인가능

q 종료

ps 실행되고 있는 프로그램들 표시.

아무 옵션을 주지 않으면 현재 터미널에서 수행하는 작업만 표시.

JOB ID라는 번호를 이용해 kill 명령 사용 가능

-ef 현재 터미널 뿐 아니라 컴퓨터에서 돌고있는 전체 작업 표시

kill [signal] [JOB ID] 필요 없는 작업 강제종료

작업을 죽일 때 “kill [JOB ID]”로 죽여보고, 실패하면 “kill -KILL [JOB ID]”로 죽인다.

Ctrl+c 실행 중 프로그램 강제종료

Ctrl+z 실행 중 프로그램 일시정지

bg 정지된 프로그램을 background로 실행

[example]

$qchem file1.in file1.out

(Ctrl+z)

$bg

$ps

출력형태

	PID	TTY	TIME	CMD
	####	pts/2	00:00:00	qchem
	%%%%	pts/2	00:00:00	serial.csh
	@@@@	pts/2	00:00:00	qcprog.exe

의 형태로 출력된다.

$kill #### %%%% @@@@

qchem이 시작한 세가지 sequence의 프로그램 강제종료

*ABOUT VI

리눅스에서는 notepad와 같은 에디터로서 vi를 사용한다.

vi A 파일 A를 vi로 연다.

:q	vi 종료
:w	현재까지 바뀐 내용을 저장
:q!	저장하지 않고 종료
:wq	저장하고 종료
:0	가장 첫줄로 이동
:gg	첫 번째로 이동
:23	23번째 줄로 이동(임의의 숫자 가능)
:G	가장 뒷줄로 이동
i	insert mode, 입력 시작(나가기 : Esc)
v	블록 모드, 글자마다 선택 가능
V	블록 모드, 줄단위로 선택 가능
Ctrl-v	수직 블록 모드
y	copy, Ctrl+C와 같은 기능
d	cut, Ctrl+X와 같은 기능
dd	한줄 지우기 (현재줄)
yy	한줄 복사 (현재줄)
p	paste, Ctrl+V와 같은 기능
P	paste, p와 복사하는 위치가 다른 것을 느껴보도록
u	undo, Ctrl+Z와 같은 기능
c	change, v로 블록을 잡고 내용을 변경할 때 유용
Ctrl+r	redo, Ctrl+Y와 같은 기능
.	macro, 직전에 수행한 변경내용을 반복
Ctrl+b	page up과 같은 기능
Ctrl+f	page down과 같은 기능
/	search. 다음 찾기 : n / 이전 찾기 : N
:new filename	filename을 불러와 동시작업 가능

Ctrl+ w+ w 화면간 이동
Ctrl+ w+ n vi 상태에서 새vi만들기
:set nu 줄번호 보기
:set nonu 줄번호 없애기

*ABOUT CODING

C로 코딩을 하려면 소스 파일 이름을 보통 *.c로 선언해준다.

ex) vi sin.c

컴파일러는 다음과 같이 선언된다.

cc -o A B B를 컴파일하여 A라는 실행파일을 만든다.
이 때 실행파일은 보통 *.x로 선언한다.
-l library를 컴파일에 포함시켜야 할 때 사용한다.
-I 헤더파일을 컴파일에 포함시켜야 할 때 사용한다.
리눅스는 일반적으로 컴파일시 헤더파일을 포함하지 않는 경우가 있다.
ex) cc -o sin.x sin.c -lm
ㄴ이 때 m은 libm.a를 뜻하며, 시스템 어디엔가 이 파일은 존재한다. (시스템이 알고 있으므로 어디 있는지는 걱정할 필요 없음)

```
cc -o hello.x hello.c
cc -o sine.x sine.c -lm
```

ㄴC언어가 최초로 생겼을 때, 이는 수학용이 아니었다. 따라서, 수학 함수는 따로 라이브러리를 만들어 두고 원할 때만 사용했다. (-lm을 이용해서) 요즘은, 수학 루틴이 매우 자주 쓰이므로 경우에 따라 -lm을 쓰지 않아도 되는 컴파일러도 많다.

```
cc -o sine.x sine.c
```

/[dir]/*.x 컴파일된 실행파일을 (경로를 포함하여)실행해준다.
보통은 현재위치에 컴파일하므로 다음과 같이 사용한다.

```
./sine.x
/home/user1/sine.x
```

UNIX는 기본으로 아무 경로도 찾지 않는다. 따라서 실행파일의 위치를 반드시 알려주어야 한다.

* ABOUT TAR

tar는 tape recording이 이루어지던 시절의 연장선이다.
폴더를 포함하여 모든 내용을 압축할 수 있다.
명령어는 tar □v(z)f A의 형태로 구성된다.
이 때 □vf는 압축을 포함하지 않는 경우에 사용한다.
□vzf는 압축을 포함하는 경우 사용되며, 파일 끝에 .gz를 관용적으로 붙인다. 압축하므로 파일 크기가 달라짐을 확인 (ls -l)

tar xvf A 묶음 풀기 (eXtract)
tar cvf A B B를 A라는 이름으로 묶기 (conCatenate)
ex) tar cvf test.tar test
tar tvf A test (Test)
tar파일내부의 내용 보기. ls -l의 형태로 보인다.

tar xvzf A　　압축 풀고 묶음 풀기
　　ex) tar xvzf test.tar.gz
tar cvzf A B　B를 A라는 이름으로 압축하여 묶기
tar tvzf A　　test, tar.gz파일 보기. ls -l의 형태로 보인다.

tar -xvzf : 풀기, tar -cvzf : 묶기, tar -tvzf : 테스트(묶여있는 파일들을 보여준다.)
cf. 옵션표시 -는 생략가능하며, 압축되어 있지 않거나 압축을 하지 않을 경우는 옵션에서z를 생략 한다.
ex. filename.tar.gz (tar:묶여 있음, gz:압축되어 있음 을 의미)
ex. tar cvzf filename.tar.gz sourcefiles (여러 개의 source file들을 압축)

```
tar cvzf test.tar.gz unix-lecture
ls -l
tar tvzf test.tar.gz
rm -rf unix-lecture
ls -l
tar xvzf unix-lecture.tar.gz
ls -l
rm test.tar.gz
```

Course 3: 고급 가기 일보 직전

*ABOUT ALIAS
alias는 일종의 별명으로서, 자주 쓰는 cmd들을 축약하여 사용할 수 있게 한다.
각자의 ID로 로그인 할 때, 저장되어있는 alias를 선언하게 된다.

alias	현재 선언되어 있는 alias를 모두 본다.
alias A 'B'	cmd B를 cmd A로 사용한다. ex) alias rm 'rm -i' ㄴ파일을 지울 때 항상 물어보게 된다.
unalias A	cmd A로 선언되어 있던 alias를 제거한다.

*ABOUT SSH, SCP
Xshell을 사용하여 현재 machine에서 다른 machine으로 접속할 수 있다.
ssh는 원격접속과, scp는 ftp와 비슷한 기능을 한다.

ssh [login name]@[machine name]
login name을 ID로 사용하여 machine에 접속한다.
ex) ssh postechian@singlet4.postech.ac.kr
ex) ssh singlet4.postech.ac.kr (ID가 동일한 경우)

scp [login name]@[machine name]:[file name]
다운로드
scp [file name] [login name]@[machine name]:
업로드

-r	아래 경로까지 모두 포함하여 업/다운로드 (recursive)
-p	파일의 시간으로 보존한 상태로 업/다운로드 (preserve)
-rp	recursive and preserve

*piping
1) > file.fil: 앞의 결과 내용을 file.fil에 저장)
2) >> file.fil (>>표시 앞의 결과 내용을 file.fil에 덧붙이기)
3) stdin | exec (|표시 앞의 결과를 standard input으로 받아 그 뒤의 명령 exec을 수행)

```
cat unix-lecture/ccsdt2.in
head -n 20 unix-lecture/ccsdt2.in
head -n 20 unix-lecture/ccsdt2.in | tail -n 5
       (결과적으로 ccsdt2.in의 16~20번째 line출력)
head -20 unix-lecture/ccsdt2.in | tail -3 > filename1
       (18 19 20 행을 출력해 filename1에 저장한다)
```

*man : 각종 프로그램의 매뉴얼을 보여준다. 필요한 옵션기능을 확인할 수 있다.

```
   man ls
   man grep
```

Course 4: 고급

* UNIX 명령만으로 히스토그램 만들기

```
grep ATOM final.pdb | cut -c 39-44 | sort -n | uniq -c
```

* shell 환경 변수 설정 (set, setenv)

```
setenv MY_ENV "beautiful day"
echo $MY_ENV
set my_var="beautiful week"
echo $my_var
setenv        : 설정된 모든 내용 프린트
set           : 설정된 모든 내용 프린트
```

특별한 의미를 지니는 환경 변수: $path, $PATH, $LD_LIBRARY_PATH

* 여러 파일에 한 작업하기: foreach

* 파일의 일괄 수정:

```
foreach f (file1 file2 file3 ... filen)
  sed -i -e "s/xxx/yyy/g" $f
  grep -v "LINE to delete" $f > _tmp_
  mv _tmp_ $f
end
```

위의 “`sed -i -e "s/xxx/yyy/g"`”에서 “g”는 global을 의미. vi의 s 명령의 syntax 참조

* login때 자동으로 설정 만들기: .login / .cshrc / .tcshrc

```
path
alias
set
setenv
```

* source

* bash와 tcsh의 차이