오픈소스 소프트웨어 실습

Open-Source Software Lab

#5

실습 담당 조교 연락처

◆실습 조교 : 김민곤

◆연구실 : 학연산클러스터 601호

◆이메일 : phenix235@hanyang.ac.kr

메일양식: [오픈소스]학번_20190000_이름

입출력 재지정 : touch

$ 명령어 > 파일

명령어의 표준출력을 모니터 대신에 파일에 저장한다.

```
[1111222333@node1 class2]$ who > names.txt
[1111222333@node1 class2]$ cat names.txt
hci       :0           2019-04-02 22:35 (:0)
1111222333 pts/0        2019-04-04 03:24 (166.104.185.49)
```

출력 추가(redirecting to append)

$ 명령어 >> 파일

명령어의 표준출력을 모니터 대신에 파일에 추가한다.

```
[111222333@node1 class2]$ date >> names.txt
[111222333@node1 class2]$ cat names.txt
hci       :0           2019-04-02 22:35 (:0)
111222333 pts/0        2019-04-04 03:24 (166.104.185.49)
2019. 04. 04. (목 ) 03:32:08 KST
```

입력 재지정(input redirection)

$ 명령어 < 파일

명령어의 표준입력을 키보드 대신에 파일에서 받는다.

```
[1111222333@node1 class2]$ wc < names.txt
  3   16 135
```

문서 내 입력(here document)

$ 명령어 << 단어

…

단어

명령어의 표준입력을 키보드 대신에 단어와 단어 사이의 입력 내용으로 받는다.

```
[1111222333@node1 class2]$ wc << END
> hello!
> world count
> END
 2  3 19
```

파이프(pipe)

$ 명령어1 | 명령어2

명령어1의 표준 출력이 파이프를 통해 명령어2의 표준입력이 된다.

```
[1111222333@node1 class2]$ who | sort
1111222333 pts/0        2019-04-04 03:24 (166.104.185.49)
hci       :0           2019-04-02 22:35 (:0)
```

명령어 열(command sequence)

$ 명령어1; ...; 명령어n

나열된 명령어들을 순차적으로 실행한다

```
[1111222333@node1 class2]$ date; pwd; ls
2019. 04. 04. (목 ) 03:43:31 KST
/home/1111222333/class2
err.txt   names.txt
```

명령어 그룹(command group)

$ (명령어1; ...; 명령어n)

나열된 명령어들을 하나의 그룹으로 묶어 순차적으로 실행한다

```
[1111222333@node1 class2]$ (date; pwd; ls) > out.txt
[1111222333@node1 class2]$ cat out.txt
2019. 04. 04. (목 ) 03:44:56 KST
/home/1111222333/class2
err.txt
names.txt
out.txt
```

조건 명령어 열(conditional command)

$ 명령어1 || 명령어2

명령어1이 실패하면 명령어2가 실행되고, 그렇지 않으면 명령어 2가 실행되지 않는다

```
[1111222333@node1 class2]$ gcc test.c || echo compile error
bash: gcc: 명 령 을  찾 을  수  없 습 니 다 ...
compile error
```

조건 명령어 열(conditional command)

$ 명령어1 && 명령어2

명령어1이 성공적으로 실행되면 명령어2가 실행되고, 그렇지 않으면 명령어 2가 실행되지 않는다

```
[1111222333@node1 class2]$ gcc test.c && a.out
bash: gcc: 명 령 을  찾 을  수  없 습 니 다 ...
```

파일 이름 대치

대표문자	의미
*	빈 스트링을 포함하여 임의의 스트링을 나타냄
?	임의의 한 문자를 나타냄
[..]	대괄호 사이의 문자 중 하나를 나타내며 부분범위 사용 가능함.

```
[1111222333@node1 class2]$ ls *.c
ls: cannot access *.c: 그런 파일이나 디렉터리가 없습니다
[1111222333@node1 class2]$ ls *.txt
err.txt  names.txt  out.txt
```

명령어 대치(command substitution)

```
[lll1222333@node1 class2]$ echo 현재 시간은 `date`
현재 시간은 2019. 04. 04. (목) 03:56:09 KST
[lll1222333@node1 class2]$ echo 현재 디렉터리 내의 파일의 개수 : `ls | wc -w`
현재 디렉터리 내의 파일의 개수 : 3
```

```
[lll1222333@node1 class2]$ echo 3 * 4 = 12
3 err.txt names.txt out.txt 4 = 12
[lll1222333@node1 class2]$ echo '3 * 4 = 12'
3 * 4 = 12
[lll1222333@node1 class2]$ name=나가수
[lll1222333@node1 class2]$ echo '내 이름은 $name 현재 시간은 `date`'
내 이름은 $name 현재 시간은 `date`
[lll1222333@node1 class2]$ echo "내 이름은 $name 현재 시간은 `date`"
내 이름은 나가수 현재 시간은 2019. 04. 04. (목) 03:58:28 KST
```

gcc 컴파일러

- Vi editor
  - ✓ 편집버퍼에 데이터를 넣을 때 반드시 다음 단계를 따라야 함
    1. 데이터를 쓰고 싶은 곳으로 커서를 옮겨야 한다
    2. 입력모드로 바꾸기 위한 명령을 사용한다
    3. 데이터를 입력한다
    4. 명령모드로 바꾸기 위해 ESC를 누른다
  - ✓ 편집버퍼에 데이터가 있으면 어떤 일을 하기 위하여 준비되어야 할 다양한 명령들
    - 커서를 움직이는 명령
    - 입력모드로 전환하는 명령
    - 변화를 주기 위한 명령

gcc 컴파일러

- 커서 이동하기

h	커서를 한 칸 왼쪽으로 이동
j	커서를 한 칸 아래쪽으로 이동
k	커서를 한 칸 위쪽으로 이동
l	커서를 한 칸 오른쪽으로 이동
LEFT	커서를 왼쪽으로 한 칸 이동
DOWN	커서를 아래쪽으로 한 칸 이동
UP	커서를 위쪽으로 한 칸 이동
RIGHT	커서를 오른쪽으로 한 칸 이동
BACKSPACE	커서를 왼쪽으로 한 칸 이동
SPACE	커서를 오른쪽으로 한 칸 이동

gcc 컴파일러

- 커서 이동하기

-	커서를 이전 줄의 처음으로 이동
+	커서를 다음 줄의 처음으로 이동
0	커서를 현재 줄의 맨 앞으로 이동
$	커서를 현재 줄의 끝으로 이동
^	커서를 현재 줄의 첫 글자로 이동(탭이나 공백이 아닌)
w	커서를 다음단어의 첫 글자로 이동
e	커서를 다음단어의 끝 글자로 이동
b	커서를 이전단어의 첫 글자로 이동

gcc 컴파일러

- 커서 이동하기

W	w와 동일, 문장 부호(“”) 무시
E	e와 동일, 문장 부호(“”) 무시
B	b와 동일, 문장 부호(“”) 무시
)	다음문장의 처음으로 이동
(	이전문장의 처음으로 이동
}	다음문단의 처음으로 이동
{	이전문단의 처음으로 이동
H	커서를 화면 맨 위로 이동
M	커서를 중간으로 이동
L	커서를 화면 맨 아래로 이동

gcc 컴파일러

- 입력모드로 전환

i	입력모드로 전환, 커서 위치 앞에서 삽입
a	입력모드로 전환, 커서 위치 뒤에서 삽입
I	입력모드로 전환, 현재 줄의 앞에서 삽입
A	입력모드로 전환, 현재 줄의 뒤에서 삽입
o	입력모드로 전환, 현재 줄의 아래에 전개
O	입력모드로 전환, 현재 줄의 위에 전개

gcc 컴파일러

- vi의 고치기 명령 :

r	단지 한 글자만 변경(입력모드로 바뀌지 않음)
R	입력하는 대로 겹쳐 써서 변경
s	삽입에 의한 한 단어의 변경
C	커서의 위치로부터 줄 끝까지 삽입에 의한 변경
cc	전체 줄을 삽입에 의해 변경
S	전체 줄을 삽입에 의해 변경

gcc 컴파일러

- 삭제 및 복사

x	커서위치의 1문자 삭제
X	커서위치의 왼쪽 1문자 삭제
dd	커서가 있는 행을 삭제
ndd	커서가 있는 곳부터 n 행 삭제
d$	커서의 위치에서 행 끝까지 삭제
d^	맨 앞에서 커서위치의 왼쪽까지 삭제
yy	커서가 있는 행을 복사
nyy	커서가 있는 행부터 n 행을 복사
P	현재 커서위치의 앞행에 붙여 넣기, 복사일 경우에는 윗 줄에 붙임
p	현재 커서에 붙여 넣기, 복사일 경우에는 아래 줄에 붙임

gcc 컴파일러

- 화면이동

ctrl + f	한 화면 아래로 이동
ctrl + b	한 화면 위로 이동
ctrl + d	반 화면 아래로 이동
ctrl + u	반 화면 위로 이동
ctrl + e	한 줄씩 아래로 이동
ctrl + y	한 줄씩 위로 이동

gcc 컴파일러

- 환경설정

:set number	행 번호 보이게
:set nonumber	행 번호 안보이게
:set autoindent	들여쓰기 설정
:set noautoindent	들여쓰기 제거
:set list	문단,조판부호 보기
:set nolist	문단,조판부호 안보이게
:set ignorecase	검색 시 대소문자 구별 제거
:set noignorecase	검색 시 대소문자 구별
:set all	현재 설정된 vi 모든 설정 값 보기

예제 프로그램

```
[111222333@node1 ~]$ vi hello.c
```

```c
#include <stdio.h>
int main(){
        printf("hello world\n");
}
```

```
[111222333@node1 ~]$ gcc hello.c
[111222333@node1 ~]$ ./a.out
hello world
```

예제 프로그램

```
$ vi test.c

    #include <stdio.h>
    #incldue <stdlib.h>

    int main(int argc, char*argv[])
    {
        if(argc !=2)
        {
            printf("error\n");
            exit(1);
        }
        printf("%s\n", argv[1]);
        return 0;
    }
```

```
[lll1222333@node1 class2]$ ./test names.txt
names.txt
```

예제 프로그램

```
$ cp test.c foo.c

$vi bar.c

    #include <stdio.h>
    #include <stdlib.h>

    int bar(){
        printf("%d\n", 1234);
    }
```

```
$ gcc –c foo.c
$ gcc –c bar.c
$ gcc –o baz foo.o bar.o
```

위의 세 명령을 하나로 합치면 :

```
$ gcc –o baz foo.c bar.c
```

예제 프로그램 : Makefile 만들기

```
$ vi Makefile

<target1> : <dependency1> <dependency2>
        <command1>
        <command2>
```

```
run : foo.o bar.o
        gcc –o run foo.o bar.o

foo.o : foo.c
        gcc –c foo.c

bar.o : bar.c
        gcc –c bar.c
```

```
[111122333@node1 class2]$ make
gcc -c foo.c
gcc -c bar.c
gcc -o run foo.o bar.o
```

예제 프로그램 : 확장된 Makefile

```
$ vi main.c

    #include <stdio.h>
    #include <stdlib.h>
    int main(void){
        printf("==== main ====\n");
        edit();
        return 0;
    }

$ vi edit.c

    #include <stdio.h>
    #include <stdlib.h>
    int edit(){
        printf("==== edit ====\n");
    }
```

```
$ vi main2.c

    #include <stdio.h>
    #include <stdlib.h>
    int main(void){
        printf("==== main2 ====\n");
        read();
        return 0;
    }

$ vi read.c

    #include <stdio.h>
    #include <stdlib.h>
    int read(){
        printf("==== read ====\n");
    }
```

예제 프로그램 : 확장된 Makefile

```
$ vi makefile

install: all
    mv edimh /home/자신학번/class2/test1
    mv readimh /home/자신학번/class2/test2
all: edimh readimh
readimh: main2.o read.o
    gcc -o readimh main2.o read.o
edimh: main.o edit.o
    gcc -o edimh main.o edit.o
main.o: main.c
    gcc -c main.c
main2.o: main2.c
    gcc -c main2.c
edit.o: edit.c
    gcc -c edit.c
read.o: read.c
    gcc -c read.c
```

```
[1111222333@node1 class2]$ make
mv edimh /home/1111222333/class2/test1
mv readimh /home/1111222333/class2/test2
```

```
[1111222333@node1 class2]$ ./test1
==== main ====
==== edit ====
[1111222333@node1 class2]$ ./test2
==== main ====
==== read ====
```

예제 프로그램 : Makefile 레이블 사용

```
$ vi makefile

OBJECTS = main.o read.o
test = $(OBJECTS)
    gcc -o test main.o read.o
main.o: main.c
    gcc -c main.c
read.o: read.c
    gcc -c read.c
clean:
    rm $(OBJECTS)
```

```
% make clean
rm main.o read.o
```

실습 과제

- ✓ io.h를 추가한 후 컴파일 하시오
- ✓ write.c를 작성한 후 컴파일 하시오
- ✓ makefile을 작성하여 컴파일 하시오

- ✓ 실행화면

```
---main.c+io.h---
---read.c+io.h---
---write.c+io.h---
```

- ✓ Makefile 작성 내용과 실행화면 캡쳐(총2개)해서 양식 맞춰 메일

실습 과제

```
$ vi io.h

    #define a "io.h"

$ vi main.c

    #include <stdio.h>
    #include <stdlib.h>
    #include "io.h"
    int main(void){
        printf("---main.c+%s---\n",a);
        readfile();
        writefile();
        return 0;
    }
```

```
$ vi read.c

    #include <stdio.h>
    #include <stdlib.h>
    #include "io.h"
    void readfile(void){
        printf("---read.c+%s---\n",a);
    }

$ vi write.c

    #include <stdio.h>
    #include <stdlib.h>
    #include "io.h"
    void writefile(void){
        printf("---write.c+%s---\n",a);
    }
```

끝