

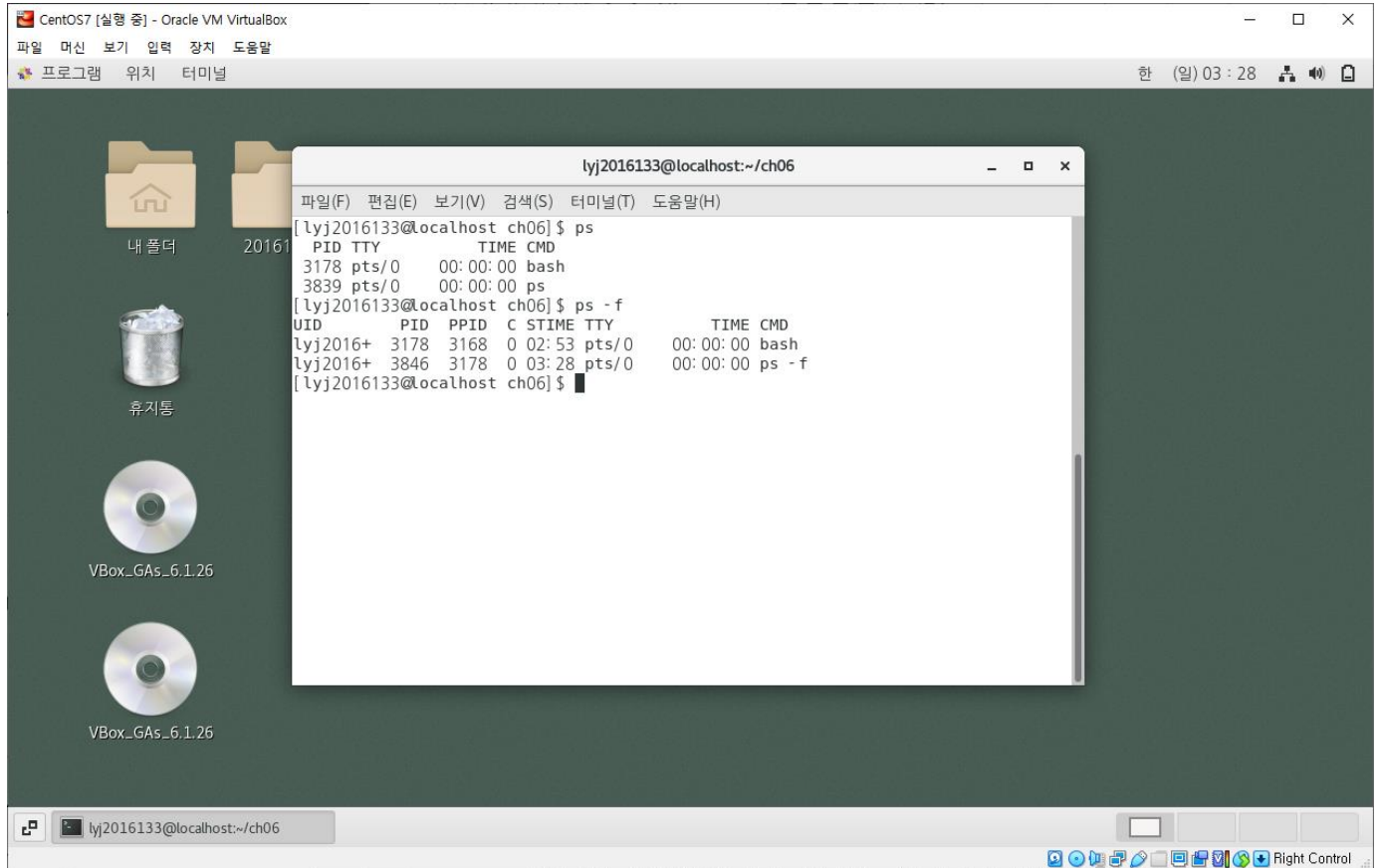
1. ps 실습

ps : process status, 즉 현재 시스템 내에 존재하는 프로세스들의 실행 상태를 요약해서 출력함.

PID=프로세스 번호, TTY=터미널 번호, TIME=CPU 실행시간, CMD=커맨드

ps -f : f옵션을 사용하면 정보가 풀 포맷으로 자세히 보임. 위의 ps명령어 단독 실행 때보다 추가된 정보인

UID=프로세스 실행자(주인), PPID=프로세스의 parent번호, C=프로세스 우선순위, STIME=start time이 있다.



ps aux : 모든 프로세스들을 보여주며, 프로세스 상태 정보가 아래의 ef옵션보다 자세함.

(a: 터미널과 연관된 프로세스 (x와 연계하여 모든 프로세스 출력),

u: (default 현재)사용자를 기준으로 정보 출력, x: 완료되지 않은 프로세스)

위에서 언급한 정보 외에 추가된 정보는 아래와 같다.

USER=프로세스 소유자 이름, %CPU=CPU 사용률, %MEM=메모리 사용률, VSZ=가상 메모리 크기,

RSS=실제 메모리 크기, STAT=프로세스 상태, START=프로세스 시작시간,

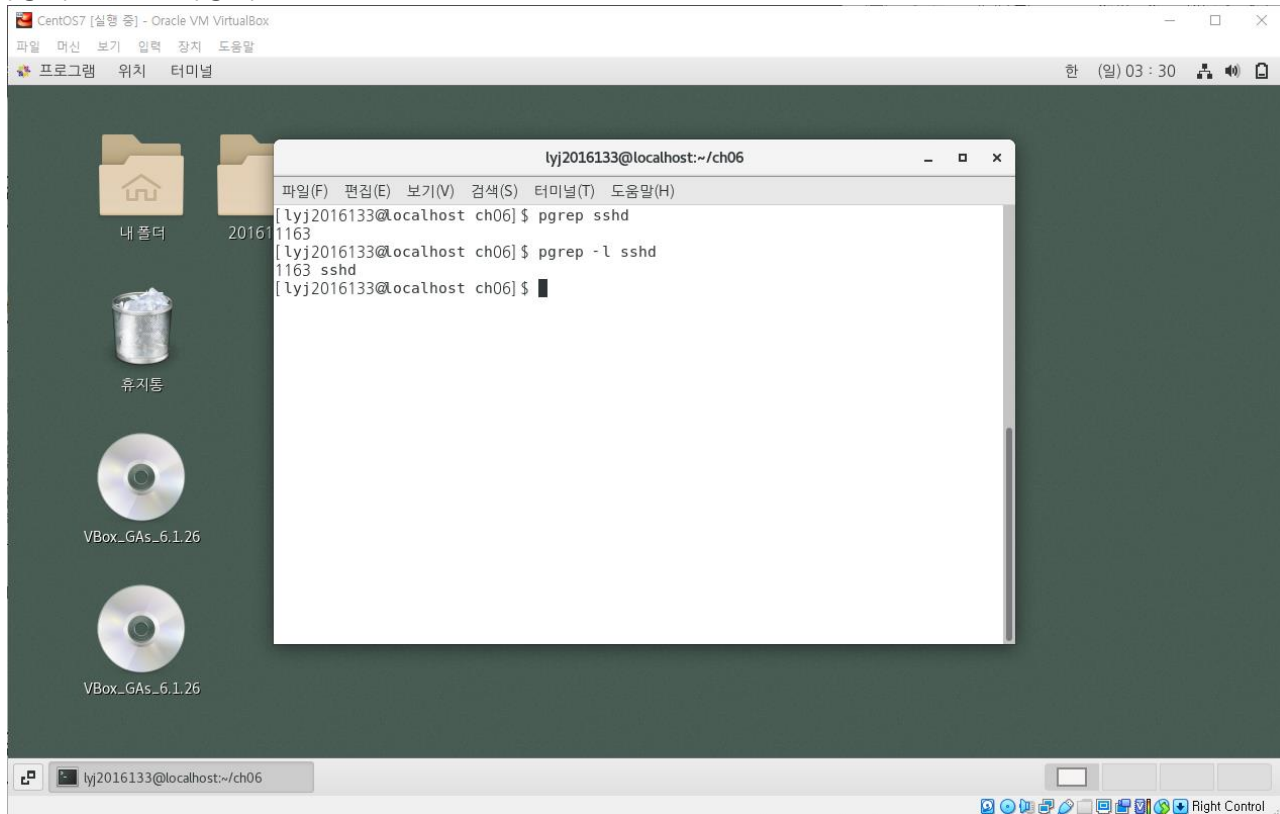
COMMAND=실행된 명령어 이름

2. pgrep 실습

pgrep sshd : 특정 프로세스만 리스트하는 pgrep. 즉 sshd라는 단어를 포함한 줄을 모두 list함.

이 때, sshd는 일종의 데몬 프로세스로(d로 끝나는 특징) 원격 로그인 서비스와 관련됨.

pgrep -l sshd : pgrep에 l 옵션을 사용함으로써 프로세스 번호 PID와 함께 프로세스의 이름을 출력함.



3. 전면처리, 후면처리 실습

sleep 100 & : 명시된 시간(초)만큼 프로세스 실행을 중지시키는 명령어 sleep을 후면 처리, 작업번호 1번 부여됨.

sleep 100 & : 명시된 시간(초)만큼 프로세스 실행을 중지시키는 명령어 sleep을 후면 처리, 작업번호 2번 부여됨.

jobs : 후면으로 실행되고 있는 작업들을 리스트해 보여주는 명령어로, 작업번호, 상태, 명령어를 화면에 출력함.

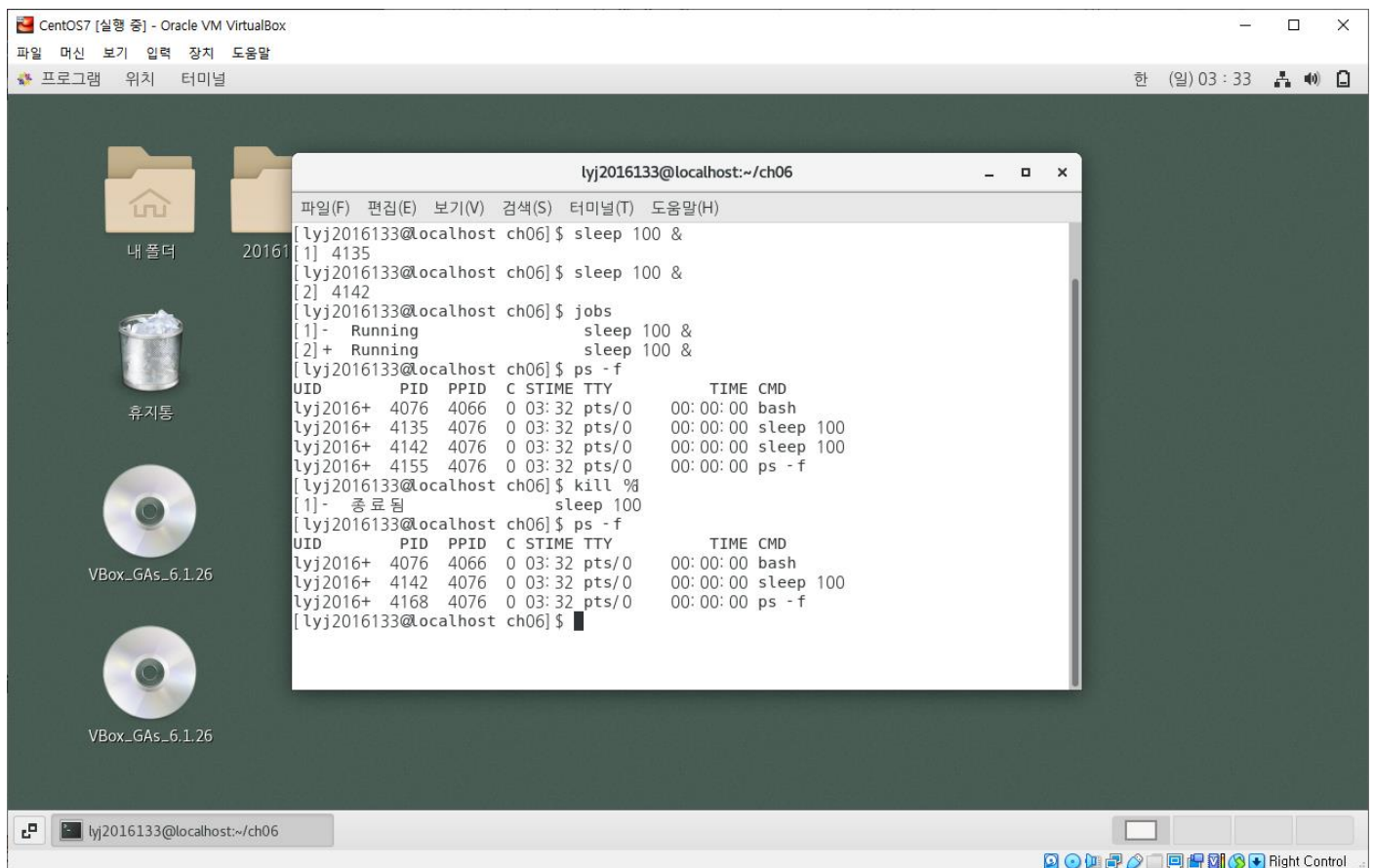
sleep 100을 두 번 후면작업처리 한 것을 볼 수 있음.

ps -f : 현재 시스템 내에 존재하는 프로세스들의 실행 상태를 보여주는 명령어 ps에 자세히 보여주는 옵션 f를 사용했음. 후면 처리한 sleep 100 두 개를 포함한 프로세스들을 확인할 수 있음.

kill %1 : 프로세스 강제 종료 명령어 kill을 사용함. 작업번호 1번을 %1과 같은 형식으로 선택하여 강제 종료함.

ps -f : 현재 프로세스 실행 상태를 보여주는 명령어 ps와 자세히 보여주는 옵션 f를 이용하여, 위에서 kill한 sleep 100 한 개가 사라진 것을 확인할 수 있음.

부모-자식관계는 PID와 PPID를 통해 확인할 수 있는데, PPID가 부모 프로세스의 번호를 나타내기 때문이다. 실행 사진을 보면 bash셸의 PID가 4076, 그리고 그 아래의 sleep 100 두개와 ps -f의 PPID가 4076이므로 bash셸이 이 세 프로세스의 부모 프로세스임.



```
CentOS7 [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
프로그램  위치  터미널
한 (일) 03 : 33

lyj2016133@localhost:~/ch06
[lyj2016133@localhost ch06]$ sleep 100 &
[1] 4135
[lyj2016133@localhost ch06]$ sleep 100 &
[2] 4142
[lyj2016133@localhost ch06]$ jobs
[1]-  Running                  sleep 100 &
[2]+  Running                  sleep 100 &
[lyj2016133@localhost ch06]$ ps -f
  UID      PID  PPID  C  STIME TTY      TIME CMD
lyj2016+  4076  4066  0  03:32 pts/0    00:00:00 bash
lyj2016+  4135  4076  0  03:32 pts/0    00:00:00 sleep 100
lyj2016+  4142  4076  0  03:32 pts/0    00:00:00 sleep 100
lyj2016+  4155  4076  0  03:32 pts/0    00:00:00 ps -f
[lyj2016133@localhost ch06]$ kill %1
[1]-  종료됨                  sleep 100
[lyj2016133@localhost ch06]$ ps -f
  UID      PID  PPID  C  STIME TTY      TIME CMD
lyj2016+  4076  4066  0  03:32 pts/0    00:00:00 bash
lyj2016+  4142  4076  0  03:32 pts/0    00:00:00 sleep 100
lyj2016+  4168  4076  0  03:32 pts/0    00:00:00 ps -f
[lyj2016133@localhost ch06]$
```

4. 작업 제어 실습

(echo START1; sleep 30; echo DONE1) & : 괄호로 명령어를 grouping하여 후면 처리함. START1을 화면에 출력, 30초만큼 프로세스를 일시정지한 다음 DONE1을 화면에 출력하라는 것. 작업번호 1번으로 지정되었음을 알 수 있음.

fg %1 : 후면 작업을 전면으로 전환하는 명령어 fg를 이용하여 1번 작업을 전면 전환함.

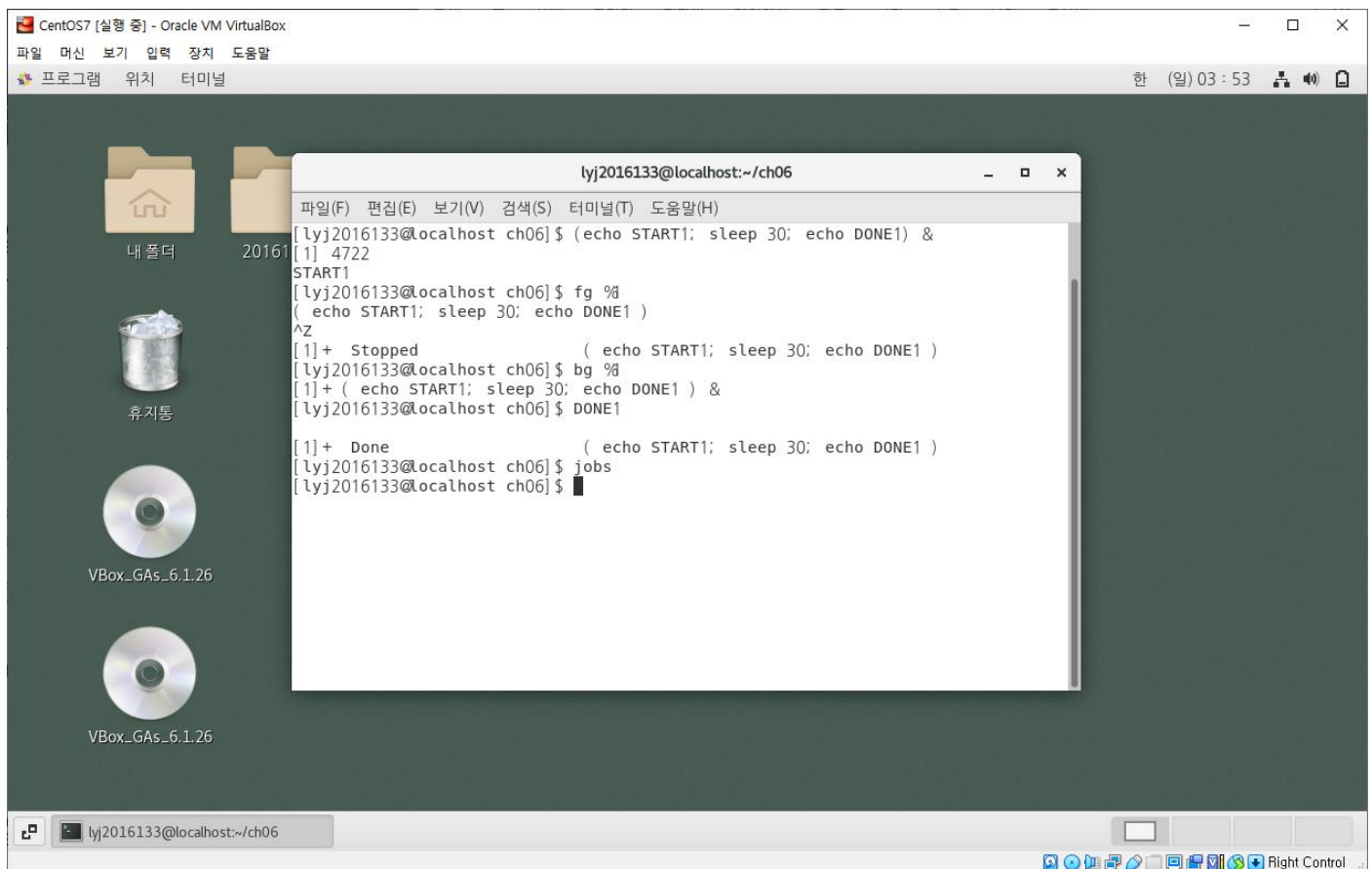
^Z : ctrl+z는 프로세스를 실행 중지하는 명령어이다. 1번 작업이 'Stopped'라고 표시되는 것을 확인할 수 있음.

bg %1 : 전면 작업을 후면으로 전환하는 명령어 bg를 이용하여 1번 작업을 후면 전환함.

이 때 잠시 기다리면 DONE1이 출력되는데, 맨 처음 괄호로 묶인 명령어를 입력한 즉시 START1이 출력되었고, 그 다음 sleep 30 명령어를 실행하느라 시간차를 두고 DONE1이 출력된 것.

엔터를 눌러 줄 바꾸면 해당 작업이 모두 완료되었음을 보임.

jobs : 후면으로 실행되고 있는 작업들을 리스트해 보여주는 명령어로, 작업번호, 상태, 명령어를 화면에 출력함. 입력했던 명령어들이 모두 실행된 후라 아무것도 출력되지 않음.



```
CentOS7 [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
프로그램  위치  터미널
한 (일) 03 : 53

lyj2016133@localhost:~/ch06
[1] 4722
START1
[lyj2016133@localhost ch06]$ fg %1
( echo START1; sleep 30; echo DONE1 )
^Z
[1]+  Stopped                  ( echo START1; sleep 30; echo DONE1 )
[lyj2016133@localhost ch06]$ bg %1
[1]+ ( echo START1; sleep 30; echo DONE1 ) &
[lyj2016133@localhost ch06]$ DONE1

[1]+  Done                    ( echo START1; sleep 30; echo DONE1 )
[lyj2016133@localhost ch06]$ jobs
[lyj2016133@localhost ch06]$
```


5. 작업제어 실습

(echo START1; sleep 50; echo DONE1) & : 괄호로 명령어를 grouping하여 후면 처리함. START1을 화면에 출력, 50초만큼 프로세스를 일시정지한 다음 DONE1을 화면에 출력하라는 것. 작업번호 1번으로 지정되었음을 알 수 있음.

(echo START2; sleep 50; echo DONE2) & : 괄호로 명령어를 grouping하여 후면 처리함. START2를 화면에 출력, 50초만큼 프로세스를 일시정지한 다음 DONE2를 화면에 출력하라는 것. 작업번호 2번으로 지정되었음을 알 수 있음.

kill -KILL 6037 : 시그널을 보내는 명령어 kill에 옵션 KILL을 사용하여 종료 시그널을 보냄.

첫 번째 프로세스 번호 6037을 입력하여 첫 번째 프로세스를 종료함. '죽었음'이라고 표시되며 해당 작업은 더 이상 실행되지 않음.

kill -STOP 6045 : 시그널을 보내는 명령어 kill에 옵션 STOP을 사용하여 중지 시그널을 보냄.

두 번째 프로세스 번호 6037을 입력하여 두 번째 프로세스를 중지함. 'Stopped'라고 표시됨.

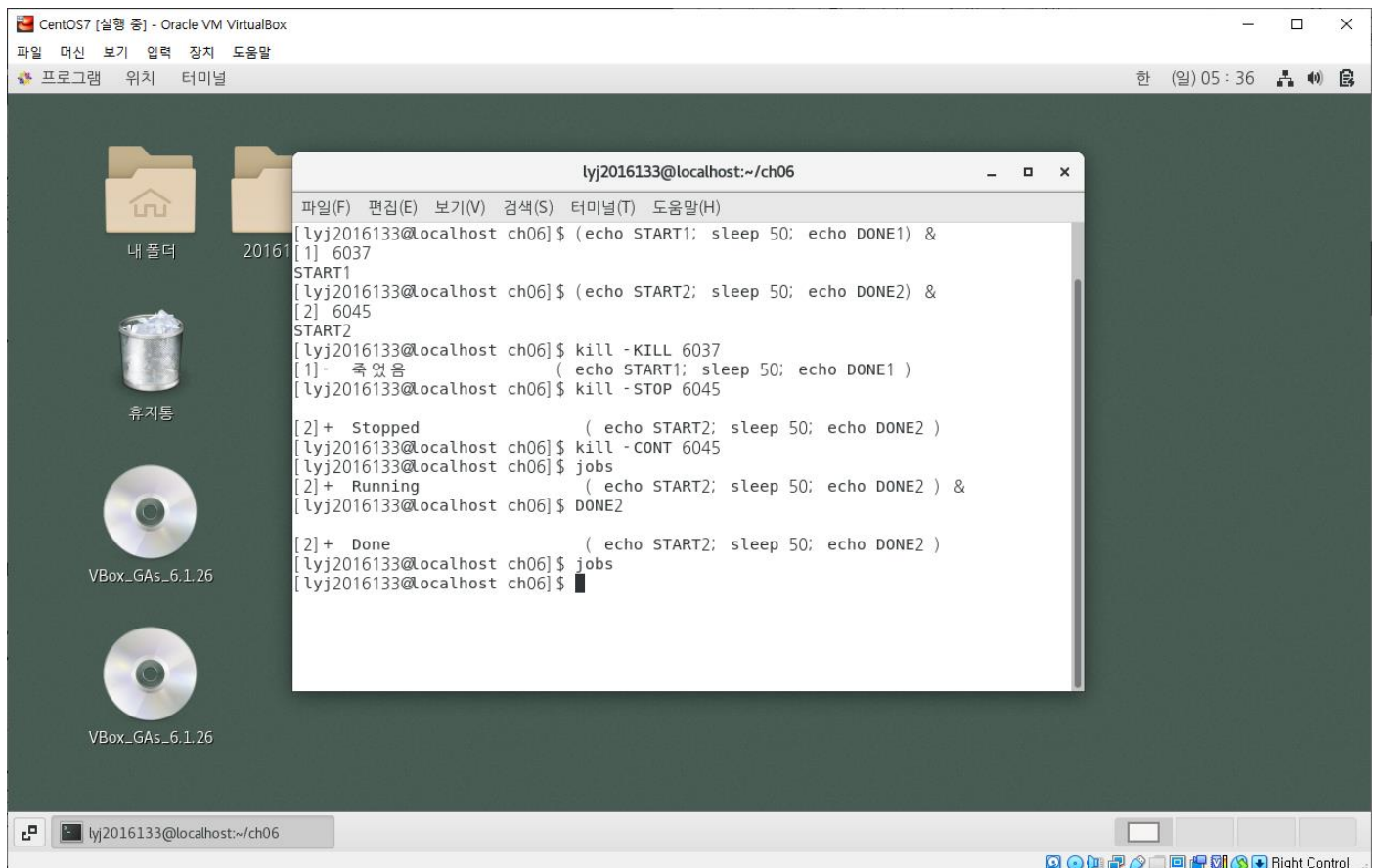
kill -CONT 6045 : 시그널을 보내는 명령어 kill에 옵션 CONT을 사용하여 실행 재개 시그널을 보냄.

두 번째 프로세스 번호 6037을 입력하여 두 번째 프로세스의 실행을 재개함. 'Running'이라고 표시됨.

jobs : 후면으로 실행되고 있는 작업들을 리스트해 보여주는 명령어로, 작업번호, 상태, 명령어를 화면에 출력함.

아래 사진에서는 두 번째 작업이 끝나기 전에 jobs를 실행한 경우 아직 2번 작업이 실행중임을 띄움.

화면에 DONE2가 echo된 다음은 입력했던 명령어들이 모두 실행된 후라 jobs를 입력해도 아무것도 출력되지 않음.



```
CentOS7 [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
프로그램  위치  터미널
한 (일) 05 : 36

lyj2016133@localhost:~/ch06
[lyj2016133@localhost ch06]$ (echo START1; sleep 50; echo DONE1) &
[1] 6037
START1
[lyj2016133@localhost ch06]$ (echo START2; sleep 50; echo DONE2) &
[2] 6045
START2
[lyj2016133@localhost ch06]$ kill -KILL 6037
[1]-  죽었음                ( echo START1; sleep 50; echo DONE1 )
[lyj2016133@localhost ch06]$ kill -STOP 6045
[2]+  Stopped                ( echo START2; sleep 50; echo DONE2 )
[lyj2016133@localhost ch06]$ kill -CONT 6045
[lyj2016133@localhost ch06]$ jobs
[2]+  Running                ( echo START2; sleep 50; echo DONE2 ) &
[lyj2016133@localhost ch06]$ DONE2

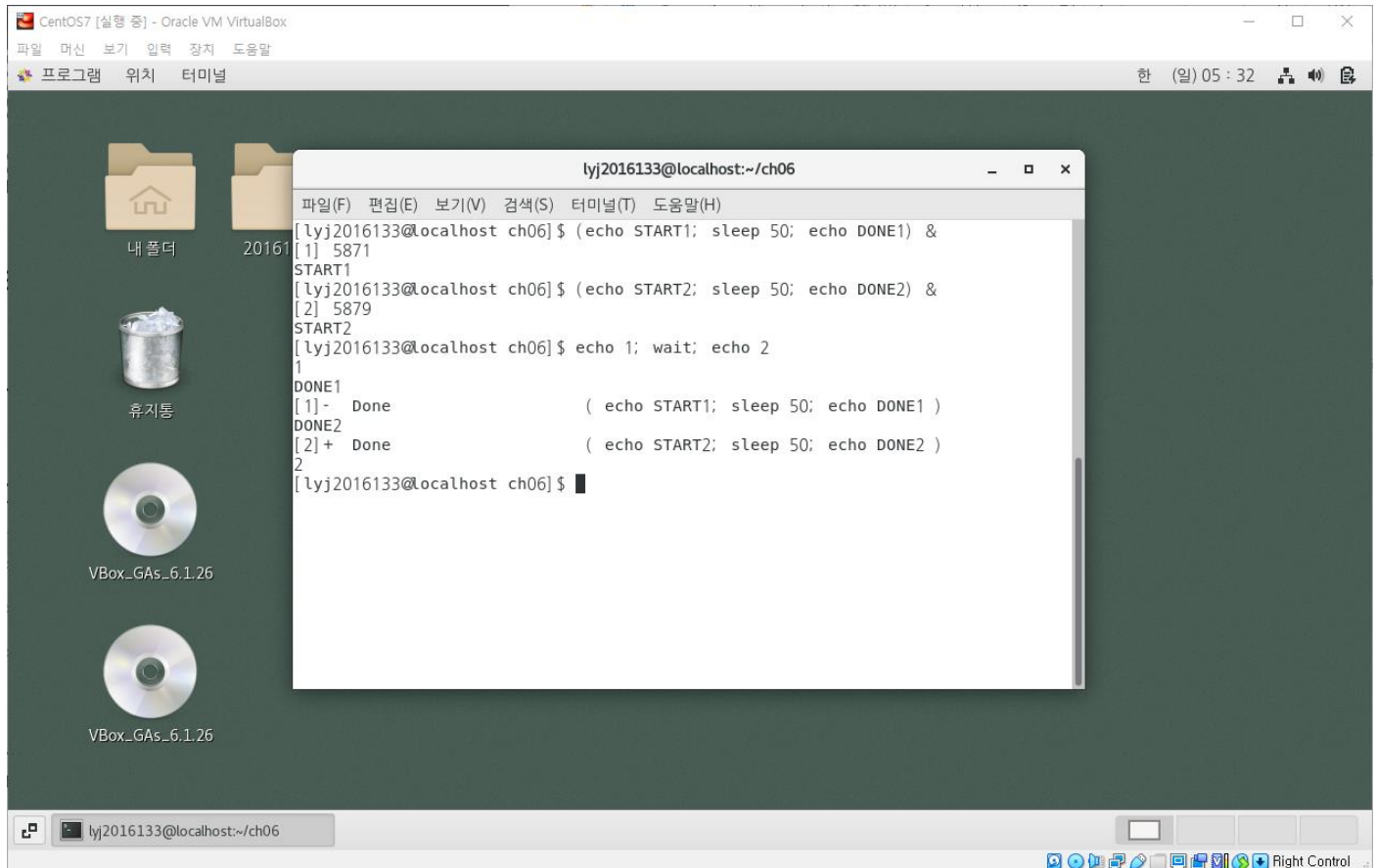
[2]+  Done                    ( echo START2; sleep 50; echo DONE2 )
[lyj2016133@localhost ch06]$ jobs
[lyj2016133@localhost ch06]$
```

6. 프로세스 기다리기 실습

(echo START1; sleep 50; echo DONE1) & : 괄호로 명령어를 grouping하여 후면 처리함. START1을 화면에 출력, 50초만큼 프로세스를 일시정지한 다음 DONE1을 화면에 출력하라는 것. 작업번호 1번으로 지정되었음을 알 수 있음.

(echo START2; sleep 50; echo DONE2) & : 괄호로 명령어를 grouping하여 후면 처리함. START2를 화면에 출력, 50초만큼 프로세스를 일시정지한 다음 DONE2를 화면에 출력하라는 것. 작업번호 2번으로 지정되었음을 알 수 있음.

echo 1; wait; echo 2 : 1을 화면에 출력한 후, 프로세스 번호를 지정하지 않은 명령어 wait를 사용했기 때문에 모든 자식 프로세스가 끝나기를 기다린다. 따라서 sleep50 후 echo DONE1, sleep 50 후 echo DONE2가 완료된 후에 echo 2가 실행되어 화면에 2가 출력된다.



```
CentOS7 [실행 중] - Oracle VM VirtualBox
파일  머신  보기  입력  장치  도움말
프로그램  위치  터미널
한 (일) 05 : 32

lyj2016133@localhost:~/ch06
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
[lyj2016133@localhost ch06]$ (echo START1; sleep 50; echo DONE1) &
[1] 5871
START1
[lyj2016133@localhost ch06]$ (echo START2; sleep 50; echo DONE2) &
[2] 5879
START2
[lyj2016133@localhost ch06]$ echo 1; wait; echo 2
1
DONE1
[1]- Done           ( echo START1; sleep 50; echo DONE1 )
DONE2
[2]+ Done           ( echo START2; sleep 50; echo DONE2 )
2
[lyj2016133@localhost ch06]$
```