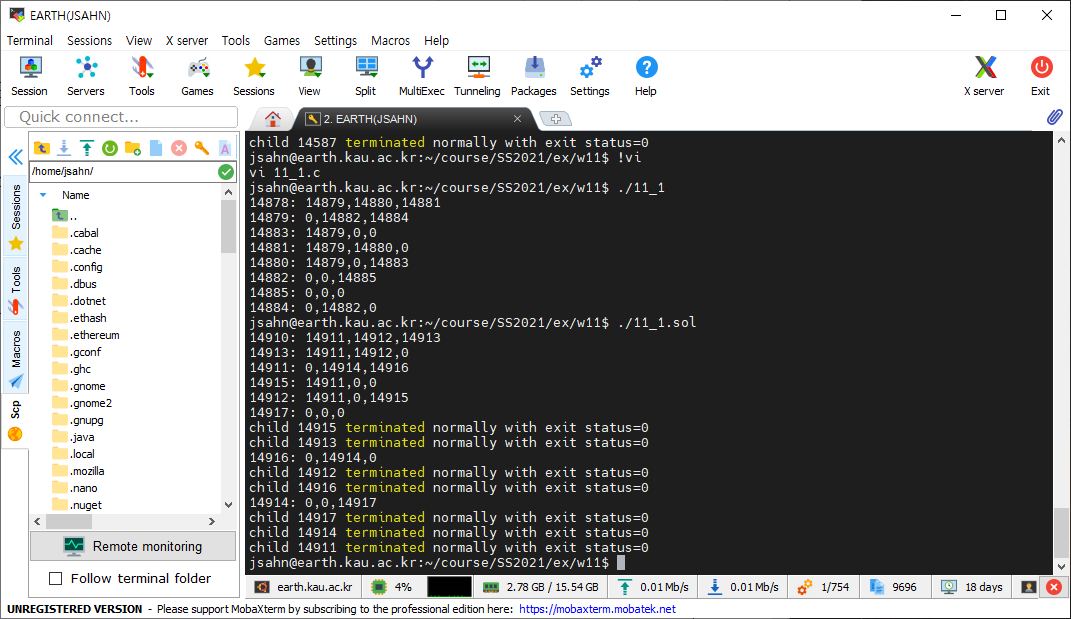
**고급시스템프로그래밍 11주차 실시간수업 과제**

|  |
| --- |
| 제출일: 2021.05.19  이름: 김현용 |

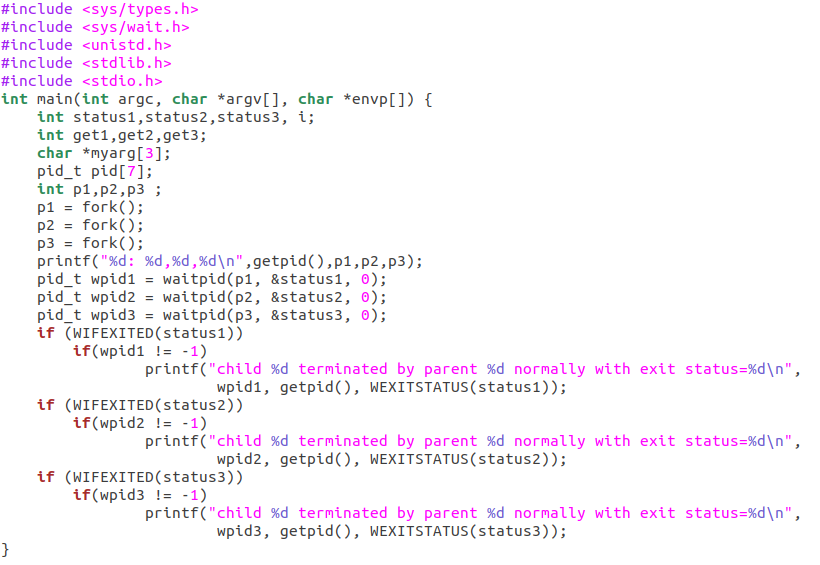
1. 다음 프로그램에 대하여 fork로 생성되는 7개 프로세스를 부모 프로세스가 리핑(reaping)하도록 코드를 추가하시오. 부모 프로세스는 자녀 프로세스를 리핑 한 후에 부모 프로세스와 자녀 프로세스 번호를 함께 출력한다.

|  |
| --- |
| #include <sys/types.h>  #include <sys/wait.h>  #include <unistd.h>  #include <stdlib.h>  #include <stdio.h>  int main(int argc, char \*argv[], char \*envp[]) {  int status;  pid\_t pid;  char \*myarg[3];  int p1,p2,p3;  p1 = fork();  p2 = fork();  p3 = fork();  printf("%d: %d,%d,%d\n",getpid(),p1,p2,p3);  } |

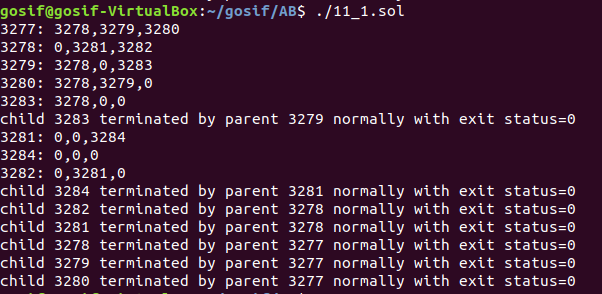
(실행화면)



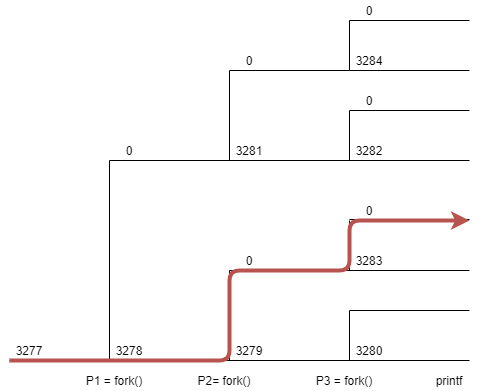
**A1-1) 코드**



**A1-2) 실행 결과**



**A1-3) 설명**

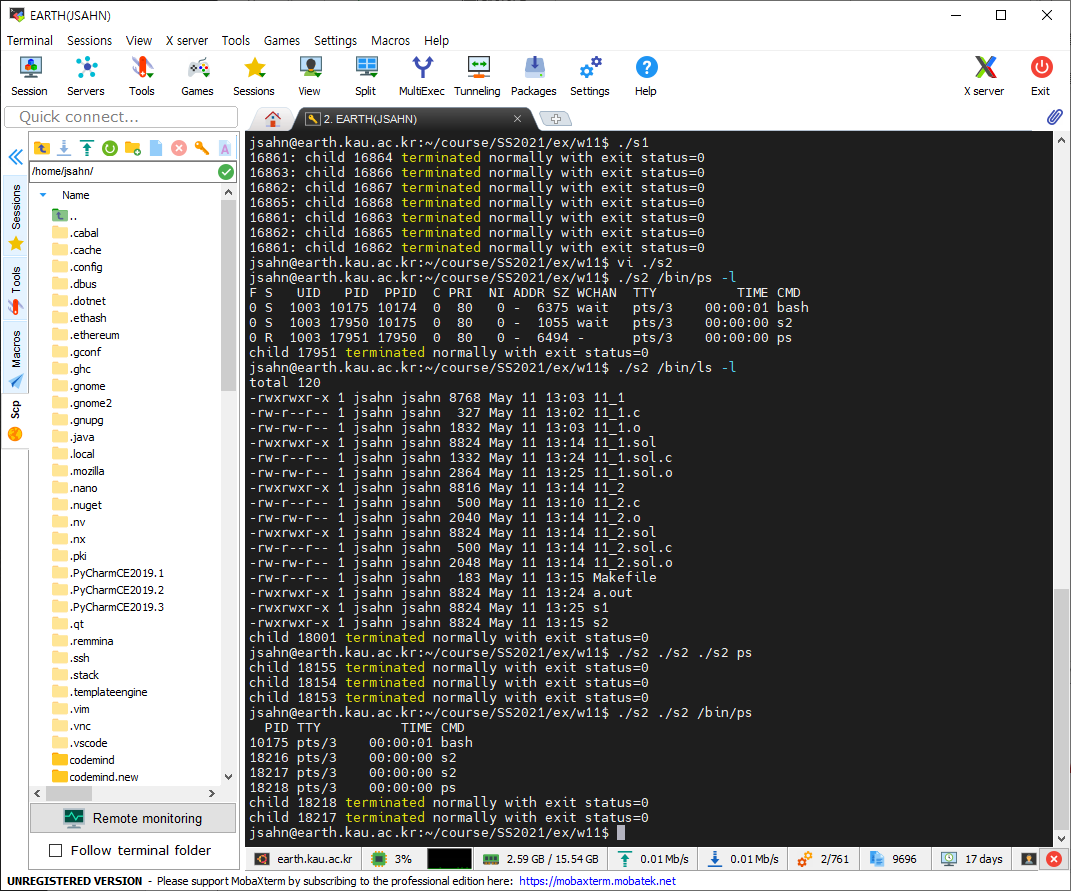


위를 보면 printf 문이 왜 저렇게 나오는지 파악 가능하다. 예를 들어, 화살표로 표시한 흐름인 3283의 경우는 p1=3278, p2=0 이었고, 자신도 child 이기 때문에 3278, 0 , 0이 출력된 것이다.

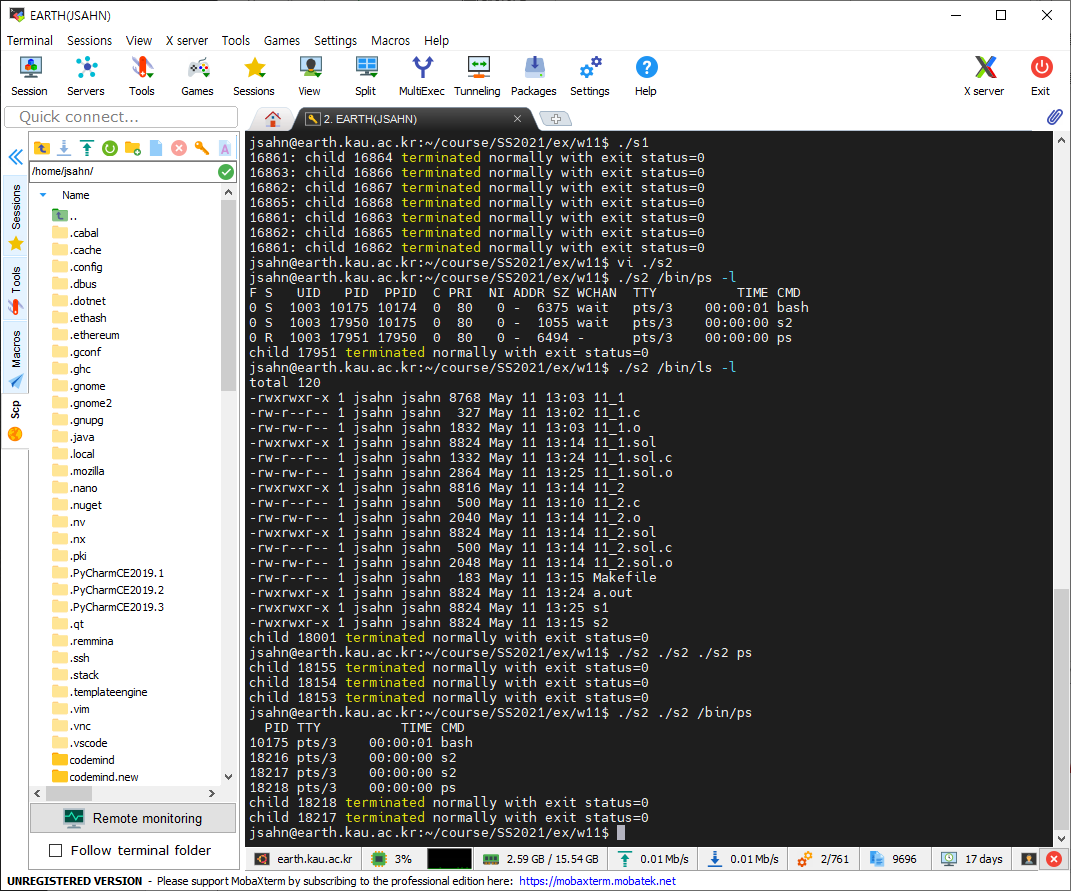
이제 waitpid 함수로 pid들을 wait 시켜줘서 부모 프로세스는 자식 프로세스가 끝날 때까지 기다리게 되고, 자식 프로세스가 종료된 상태면 printf문을 출력하게 된다. 따라서, 방금 예시의 3278의 경우는 바로 위 부모인 3279가 리핑해주는 것을 확인할 수 있다.

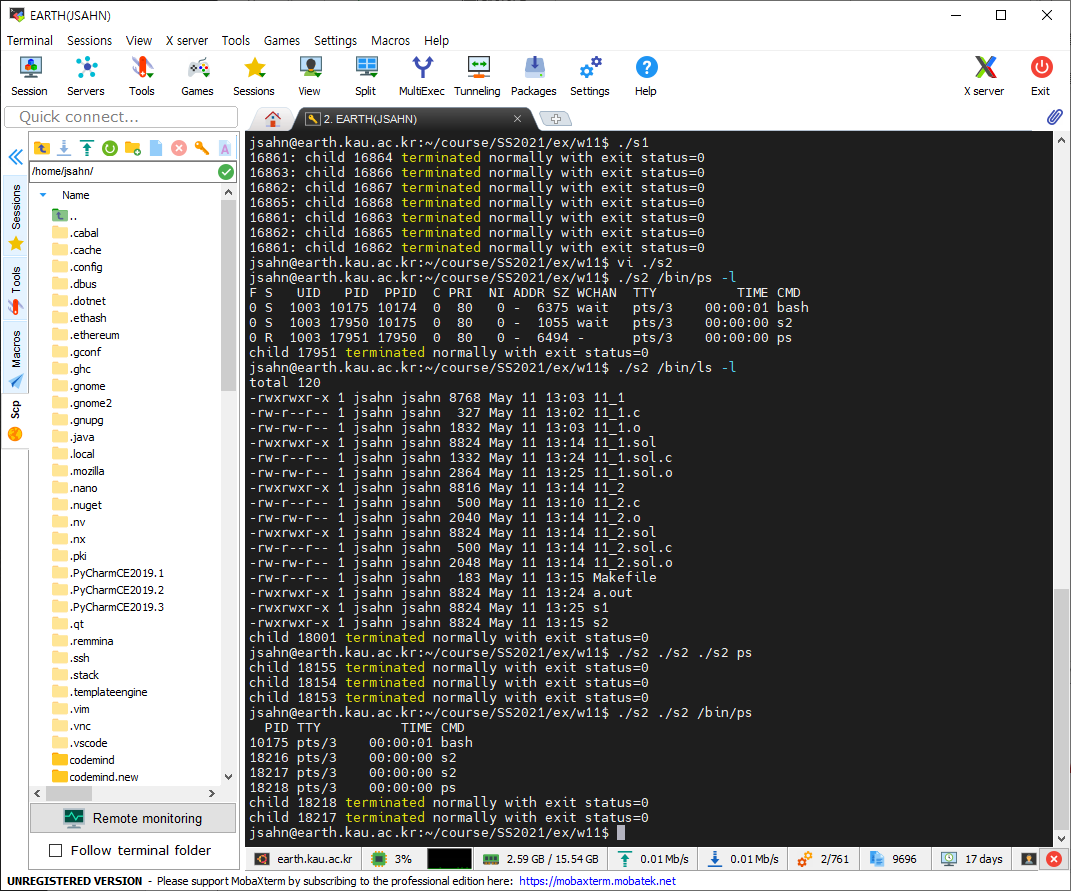
1. 실행파일 경로와 인자를 받아서 해당 코드를 수행하는 자식 프로세스를 생성하고 자식 프로세스를 리핑 하는 프로그램을 작성하시오.

(실행화면 1)

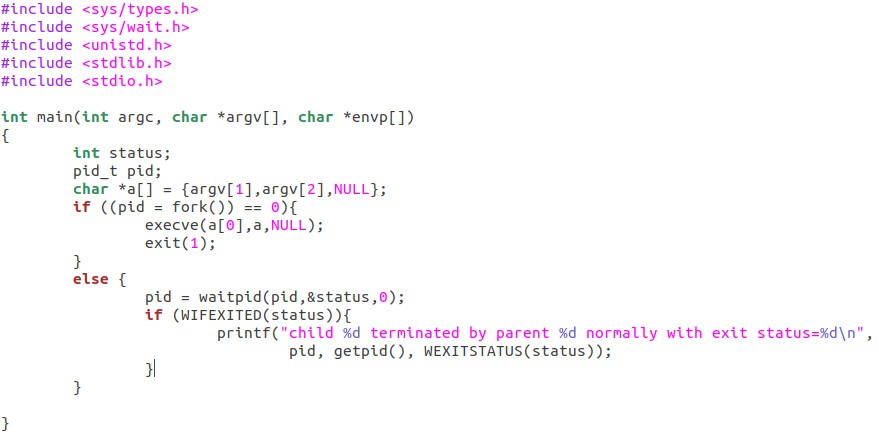


(실행화면2)

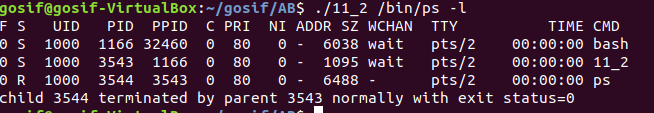


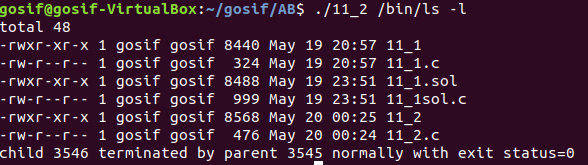
(실행화면 3)

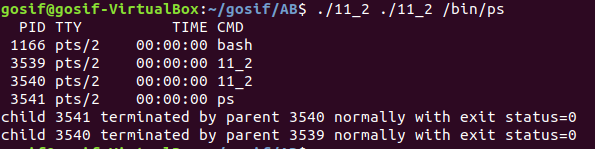
**A2-1) 코드**



**A2-2) 실행 결과**







**A2-3) 설명**

먼저 shell에 명령어를 입력하면 argv 변수에 자동으로 저장된다. 따라서 ./11\_2 /bin/ps -l 을 입력할 경우, argv[0] = “./11\_2”, argv[1] = “/bin/ps” argv[2] = “-l” 이 저장되게 된다. 그리고, execve 함수의 경우는 첫 번째 인자가 실행할 파일명이고, 두 번째 인자는 NULL까지의 명령어를 입력하게 된다. 따라서 임의로 a라는 char형 포인터배열을 생성해서, argv[0]을 제외하고 넣었고, 그에 맞게 자식 프로세스가 해당 파일의 명령어를 수행하게 하였다. 여기서 부모 프로세스는 자식 프로세스가 종료될 때까지 기다리게 한 뒤, 종료 되면 printf문을 출력하게 하였다.