***유닉스 프로그래밍 프로젝트 1 보고서***

***2019년 11월 7일***

***컴퓨터공학과***

***12171619***

***박윤수***

1. 요구사항 정의

cd와 exit, foreground & background 구현 후 테스트 진행

cd의 경우 chdir 시스템콜을 이용해서 구현하고(현재 프로세스, 즉 myshell 프로세스에 한해서), exit는 return을 이용해 구현, background는 고아 프로세스 생성의 원리를 이용해 구현한다.

1. 코드 설명

스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

위의 코드는 simple코드의 내용과 크게 다르지 않다. 알아 두어야 할 점은 makelist 함수의 반환값은 argument의 개수, 즉 argument들이 벡터형식으로 변환되었을 때의 벡터의 크기이다.

텍스트이(가) 표시된 사진

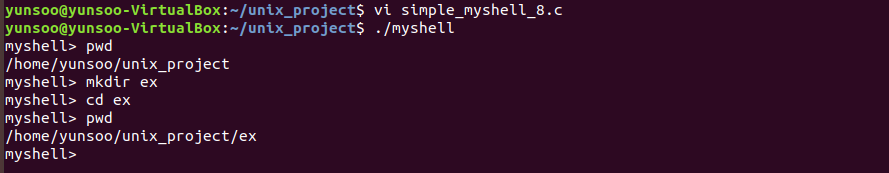
자동 생성된 설명

우선, 받은 인수들을 벡터형식으로 만들기 위해 makelist 함수를 이용한다. 만약 cmdvector의 마지막 성분이 “&”이면 백그라운드에서 실행해야 하므로 backflag를 1로 초기화하고, 벡터의 마지막 성분을 NULL로 초기화 한다.

이후, 명령어가 “exit”일 경우 break문을 통해 while 반복문을 빠져나가고 프로그램을 종료하게 되고, 명령어가 “cd”일 경우 chdir 시스템 콜을 이용해 현재 myshell 프로세스의 directory를 변경한다. 그 외의 명령어들은 모두 switch case문으로 넘어간다.

switch case문에서 fork을 실행하고 자식일 때, 부모일 때, 오류가 생길 때로 나누어서 case문을 실행한다. 여기서 pid가 0일 경우, 즉 자식의 경우 backflag가 1인지 판단하고 0일 경우 포그라운드에서 실행되야 하므로 원래의 코드대로 execvp를 실행한다. 만약 backflag가 1이면, 백그라운드에서 실행되야 하므로 자식 프로세스에서 fork를 실행하여 새로 생성된 자식(손주)이 execvp를 실행하고 기존의 자식은 종료한다.

1. 테스트

-1. 

pwd 명령어를 통해 현재 프로세스의 directory를 출력한다. 이후 ex라는 directory를 새로 만들고 cd 명령어를 통해 현재 프로세스의 directory를 ex로 옮긴다. 다시 pwd 명령어를 입력하면 현재 프로세스의 directory가 출력되는데 /home/yunsoo/unix\_project/ex 로 변경됨을 알 수 있다.

-2.



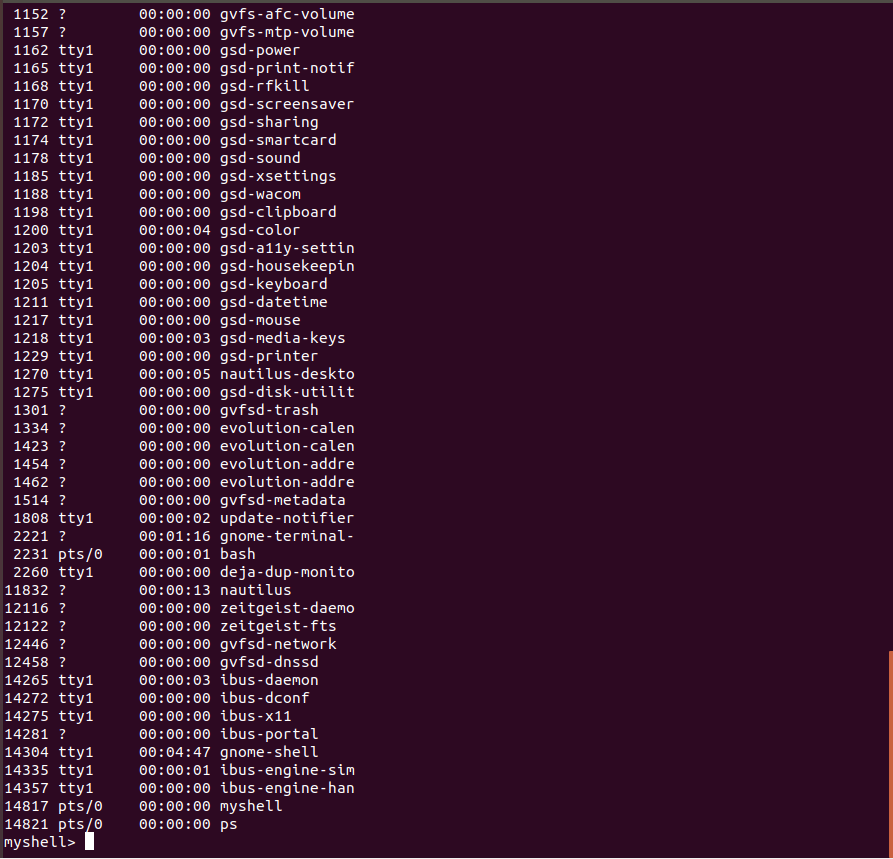
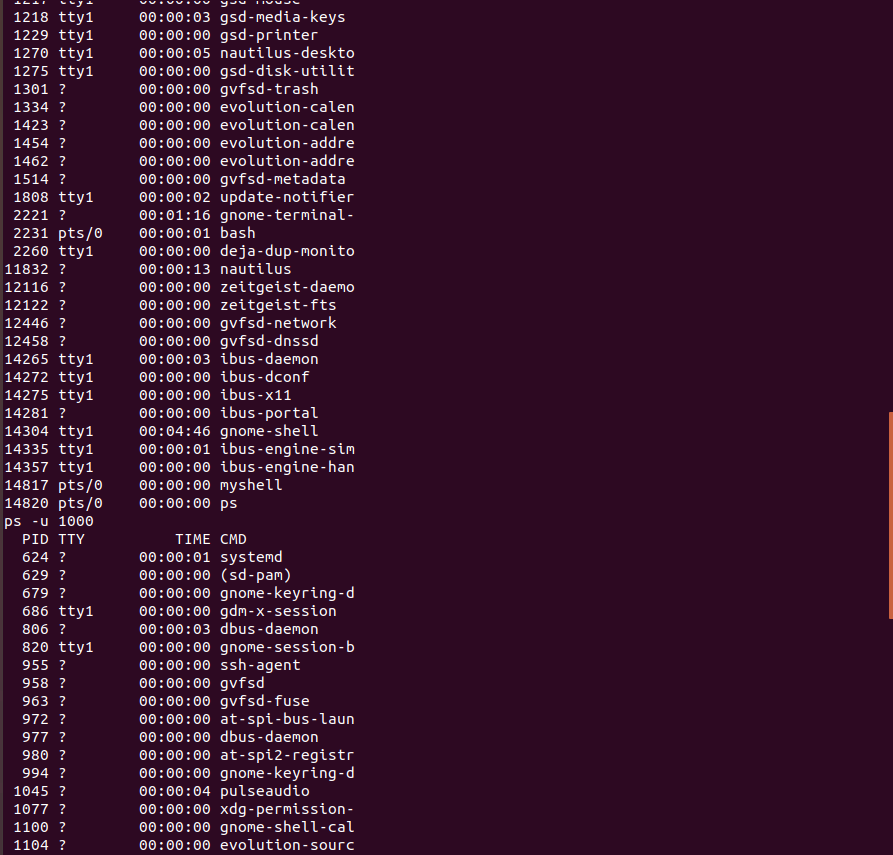
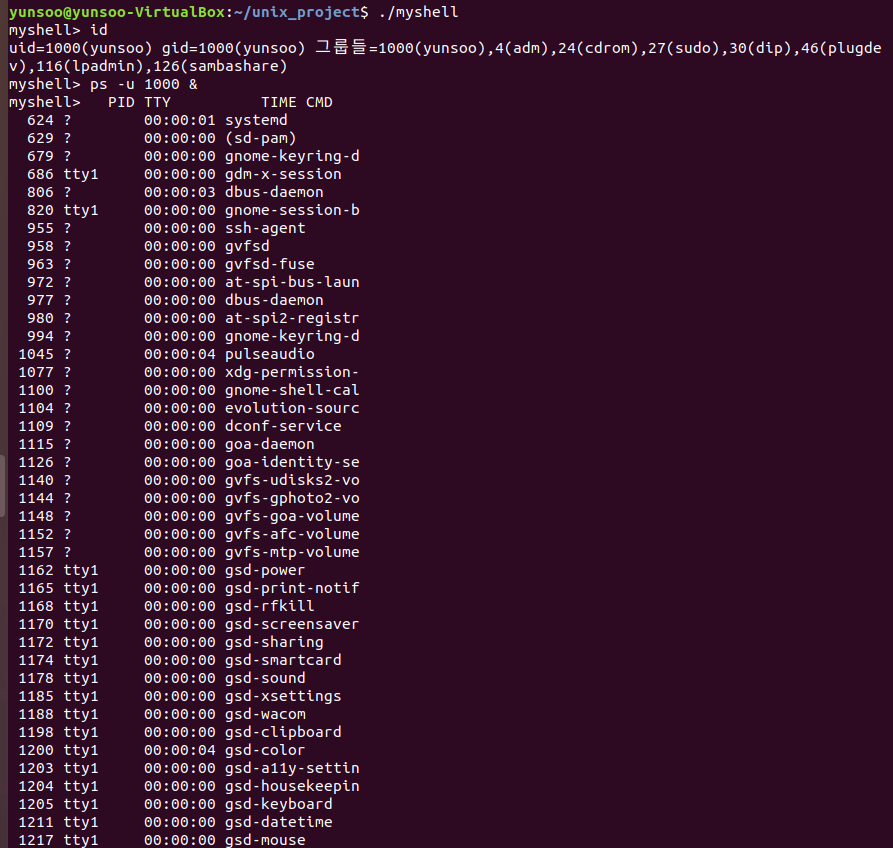
exit시 myshell이 종료된다.

-3.



먼저 sleep 10을 입력하면 10초 후에 프롬프트가 출력된다. 이후 sleep 10 &을 입력하면 sleep 10이라는 명령어를 백그라운드에서 실행하므로 프롬프트가 바로 출력된다. 백그라운드에서 실행되고 있는 프로세스를 보기 위해 ps를 입력하면 다음과 같이 sleep이 백그라운드에서 실행되고 있음을 확인할 수 있다.

-4.



루트유저로 테스트를 진행해서 ps입력 시 출력되는 프로세스가 많음에 우선 죄송합니다.

ps -u 1000 &를 입력하면 백그라운드에서 ps가 실행되는데 ps 자체가 원래 터미널 창에 출력하는 명령어이므로 포그라운드에서와 마찬가지로 프로세스 목록을 출력한다. ps가 백그라운드에서 실행되므로 먼저 프롬프트가 출력된 이후 프로세스 목록이 출력된다. (ps를 실행하는 손주 프로세스가 실행되기 전에 자식 프로세스가 먼저 종료하여 부모 프로세스에서 while문을 처음부터 다시 실행하므로 프롬프트가 먼저 출력된다.)

이후 ps -u 1000을 입력하면 포그라운드에서 ps가 실행되는데 위에서 백그라운드로 입력했을 때와 마찬가지로 프로세스 목록을 출력한다. 이번엔 ps가 포그라운드에서 실행되므로 프로세스 목록이 출력된 이후 프롬프트가 출력된다.

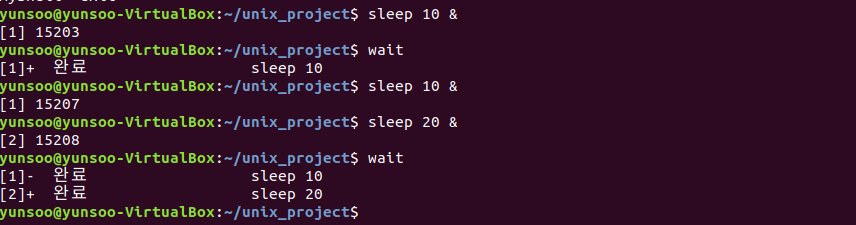
백그라운드에서 실행하는 것을 고아 프로세스가 생성되는 원리로 코딩을 했기 때문에 백그라운드에서의 실행을 마치면 그 프로세스는 부모가 wait함수를 호출하지 않았으므로 좀비가 된다. 이후 init 프로세스가 고아인 프로세스를 입양하여 wait함수를 호출함으로써 종료된다.

다만, 위의 작업에서는 백그라운드에서 실행된 프로세스가 좀비가 되지 않고 바로 종료됐는데 이유는, init 프로세스가 너무 빨리 입양하고 wait함수를 호출해서 좀비가 되는 과정을 확인하지 못하는 것 같다.

-5. 

백그라운드에서 sleep 10과 sleep 20을 실행하고 ps를 입력하면 프로세스 목록이 출력되고 이후 프롬프트가 출력된다. 즉, 백그라운드를 기다리는 테스트를 진행하는데 백그라운드 프로세스를 기다리지 않는다.

백그라운드 프로그램을 기다린 후에 프롬프트를 출력하기 위해선 ps라는 명령어를 입력하는 게 아니라 wait이라는 명령어를 입력해야 한다. wait명령어는 백그라운드에서 실행되고 있는 모든 프로세스가 종료될 때까지 기다리는 기능을 한다. wait 명령어를 구현함으로써 백그라운드 프로세스를 모두 기다린 후에 프롬프트를 출력하도록, 즉 myshell 프로세스가 실행되도록 한다.



기본 쉘에서 wait명령어를 입력했을 때의 실행 과정이다. 백그라운드를 모두 기다린 후에 프롬프트를 출력하는 것을 확인할 수 있다.