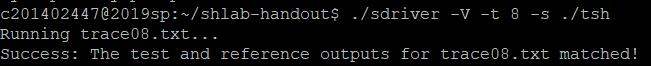
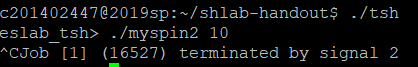
**2019 시스템 프로그래밍**

**- week08 -**

|  |  |
| --- | --- |
| **제출일자** | 2019.11.25 |
| **분 반** | 02 |
| **이 름** | 한원희 |
| **학 번** | 201402447 |

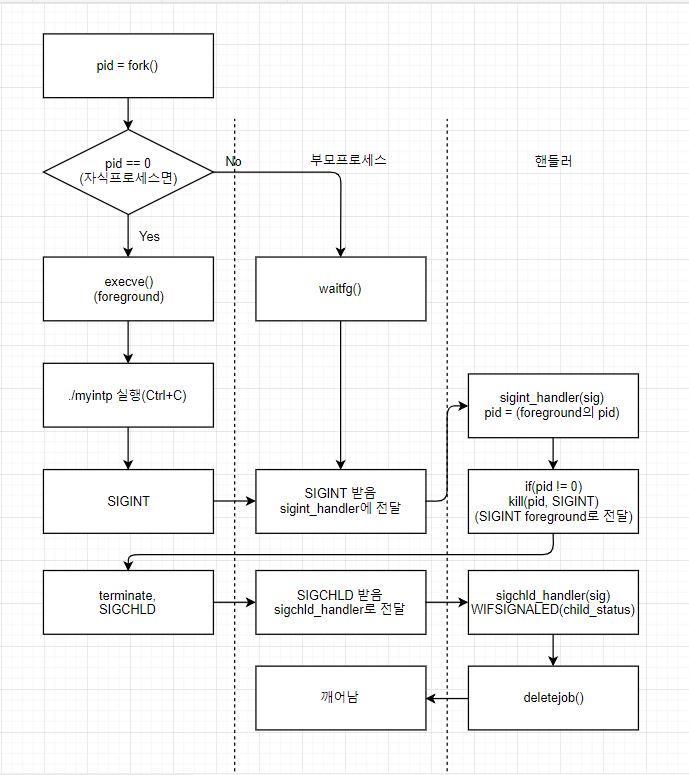
|  |
| --- |
| **Trace 번호 (08)** |



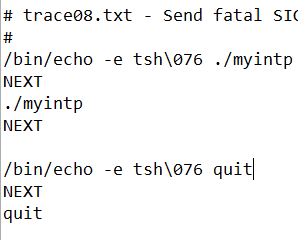


ctrl-c를 입력하면 foreground job이 종료됩니다

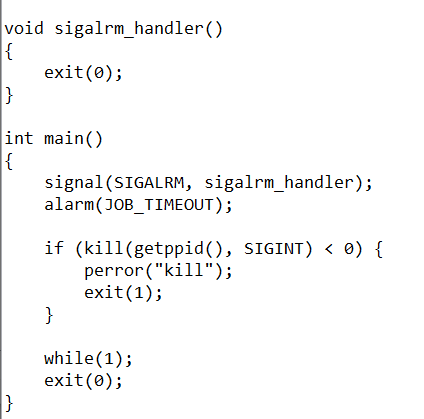
|  |
| --- |
| 08 플로우 차트 |



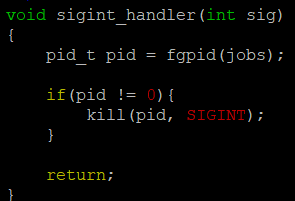
|  |
| --- |
| Trace 해결 방법 설명 |



txt 파일을 확인해보면 myintp를 호출해줍니다.

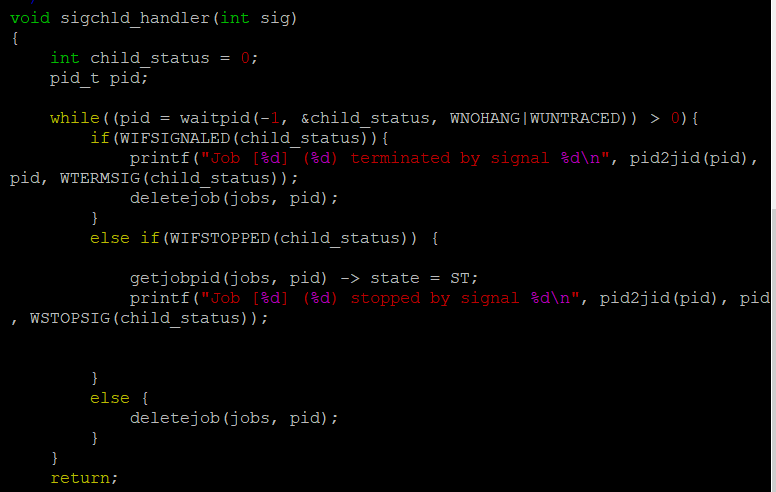


myintp를 확인해보면 kill 함수를 사용해서 sigint 신호를 부모 프로세스에게 보내주는 역할을 합니다. sigint 가 발생하면 sigint\_handler() 함수가 자동으로 호출됩니다.

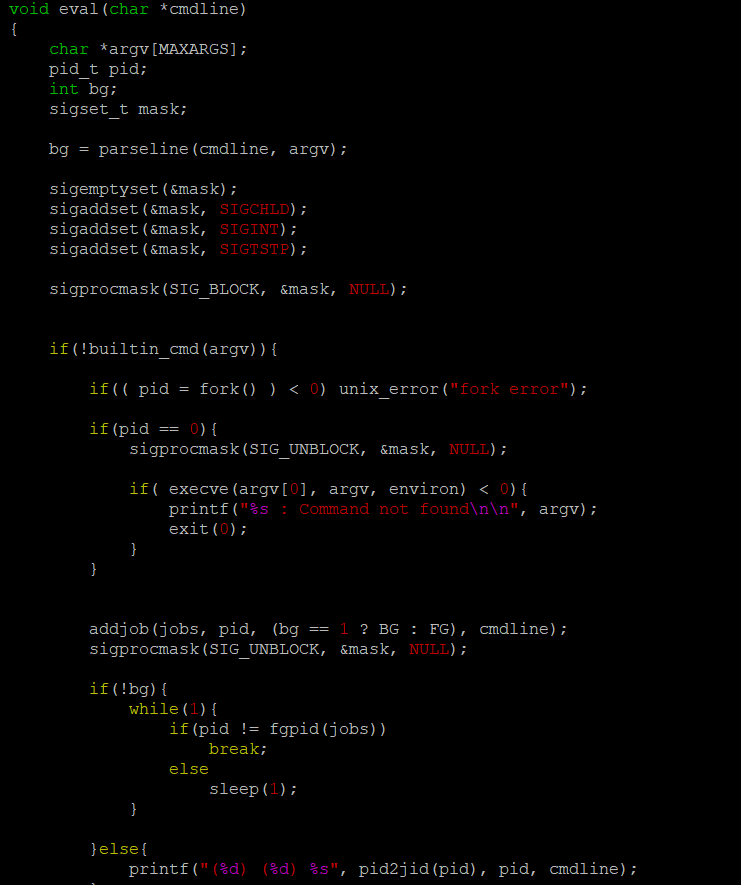


먼저 sigint\_handler를 보면 현재 프로세스의 id를 찾아, kill 함수를 이용해 프로세스에 kill 함수를 보내주는 역할입니다.

만약 프로세스가 종료디면 자식 프로세스는 부모 프로세스에게 sigchld 신호를 보내고 이는 sigchld\_handler가 처리하도록 되어있습니다.



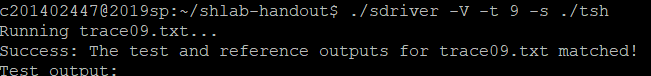
함수를 보면 waitpid를 이용해 프로세스가 종료하기를 기다렸다가 종료된 프로세스의 id를 pid에 넣어줍니다. WIFSIGNALED를 이용해 프로세스가 sig를 받아 종료된다면 프린트 해주고 deletejob을 실행합니다. 밑의 else if 문은 trace09때 사용합니다.

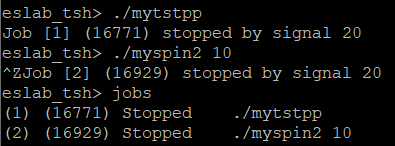


eval 함수를 확인해보면 signal들을 추가해주었고 그이후 fork를 호출하기 이전에 signal을 블록하고 자식 프로세스일때와 addjob이후에 언블록해줍니다. race를 막기 위해섭니당.

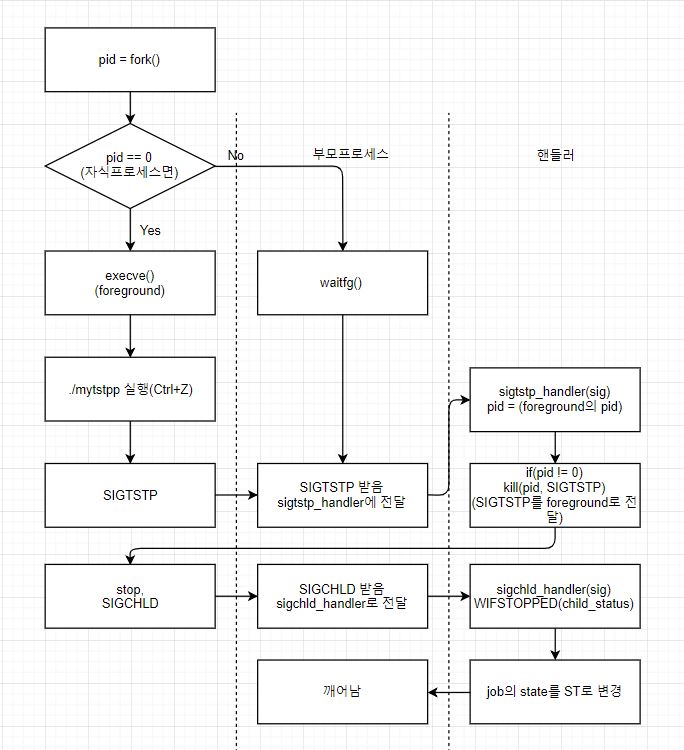
if(!bg) 밑의 부분은 부모 프로세사가 자식 프로세서의 종료를 대기 후에 처리해주는 일입니다.

|  |
| --- |
| **Trace 번호 ( 09 )** |

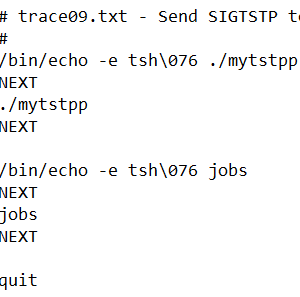




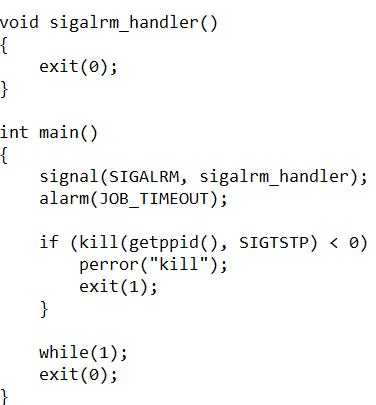
|  |
| --- |
| 09 플로우 차트 |



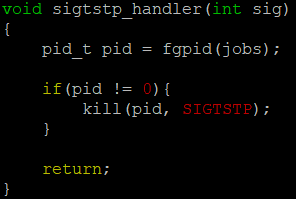
|  |
| --- |
| Trace 해결 방법 설명 |



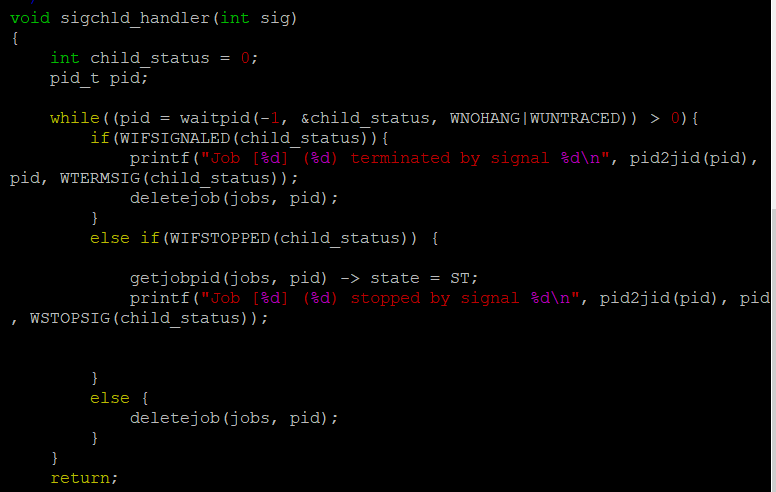
mytstpp를 호출하네요 찾아가보겠습니다.



kill 함수를 이용해 SIGTSTP 시그널을 부모 프로세스에게 보내줍니다. 메인 함수에서 ctrl-z가 입력되면 SIGTSTP 시그널을 보냅니다. 이 시그널은 핸들러에서 처리합니다.



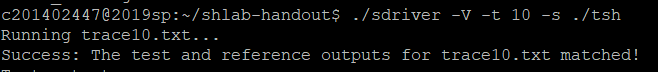
sigstp 핸들러를 보면 현재 프로세스의 id를 받아 kill 함수를 사용해 SIGTSTP 시그널을 보냅니다.



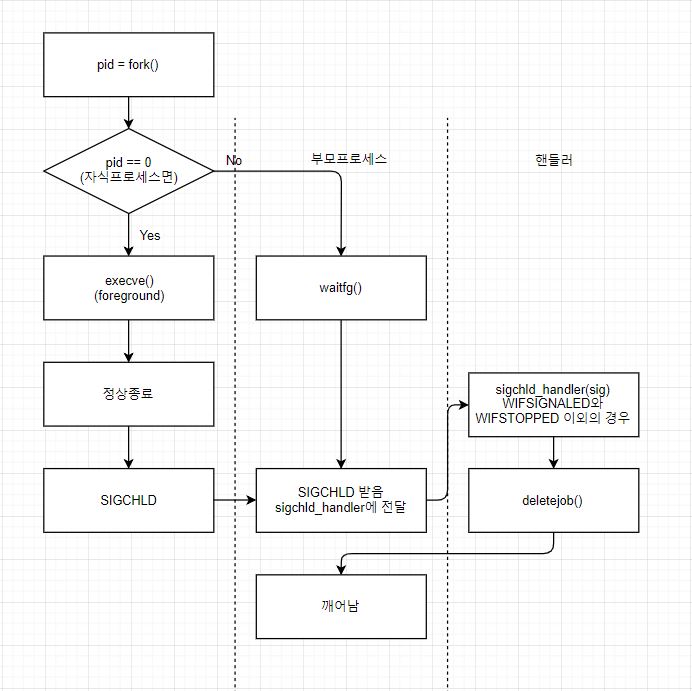
trace08과 동작이 비슷합니다. 먼저 waitpid를 수정하는데 WNOHANG|WUNTRACED는 모든 자식 프로세스들이 정지하였거나 종료하였다면 리턴값 0으로 반환합니다. 아니면 반환되거나 자식들 중 한개의 pid와 동일한 값으로 반환해줍니다.

else if 를 사용해서 프로세스가 정지 상태일때 처리해주는 코드를 추가합니다. 프로세스가 정지된 상태이면 job의 상태를 ST로 바꾸어주고 프린트문을 작성해줍니다.

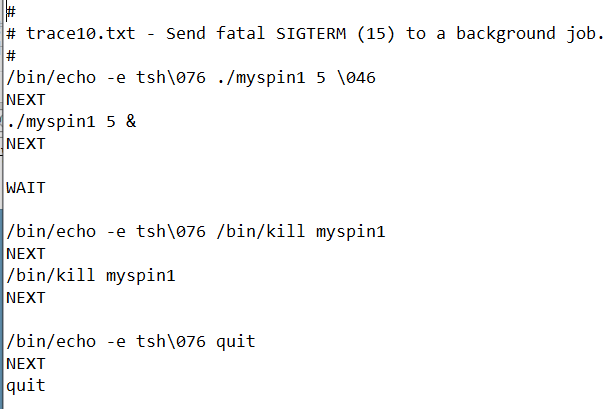
|  |
| --- |
| **Trace 번호 ( 10 )** |



|  |
| --- |
| 10 플로우 차트 |

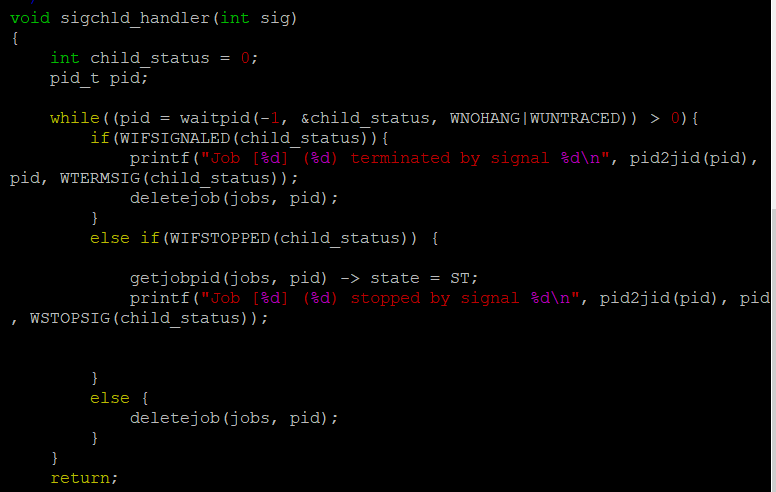


|  |
| --- |
| Trace 해결 방법 설명 |



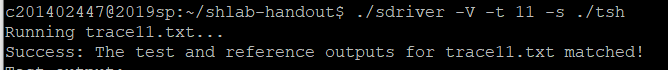
myspin1을 5초동안 실행하고 기다린다음 mypin1을 시간제한 없이 실행하고 kill함수를 이용해 종료되는것을 볼 수 있습니다.

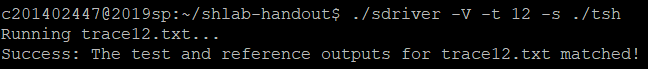
프로세스가 정상적으로 종료되면 프로세스를 없애주어야 합니다. 그리고 kill시그널이 들어왔을때에도 프로세스를 없애 주어야 합니다.- 좀비 만들기 방지 -



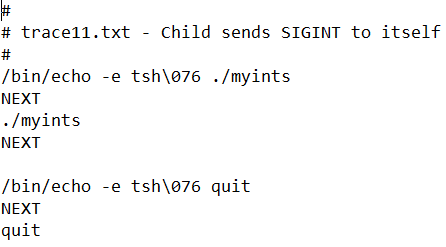
sigchld\_handler 부분에서 WIFSIGNALED 부분과 맨 마지막의 else 부분의 동작을 보면 됩니다. 시그널이 들어왔을때에 프린트를 하고 deletejob을 해주고 else부분은 프로그램이 자동적으로 종료되었을때의 부분이니까 그냥 deletejob을 해주면 됩니다. 이렇게 해서 trace10은 그냥 통과됩니다.

|  |
| --- |
| **Trace 번호 ( 11, 12 )** |

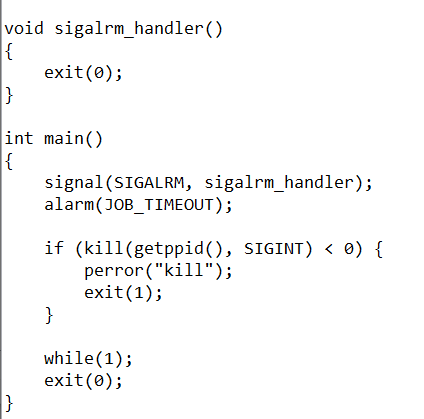




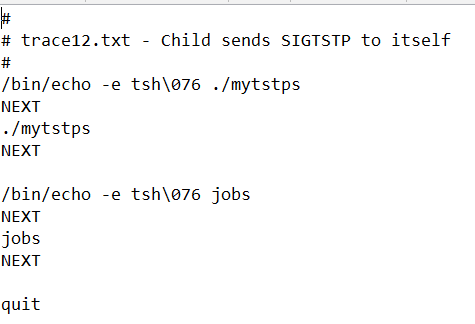
|  |
| --- |
| Trace 해결 방법 설명 |



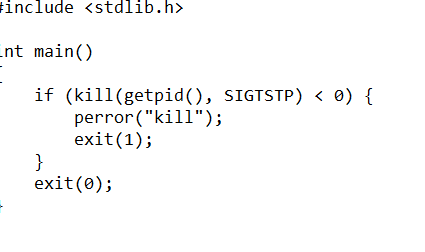
myints를 호출하는데 위의 trace08에서도 사용하였습니다. 다시 보면



kill 함수를 통해 SIGINT 신호를 보내줍니다.



12는 mytstps 를 호출합니다.



kill 함수를 이용해 SIGSTP 신호를 자식 프로세스에게 전달해줍니다.

trace08, 09 를 구현하면서 자동적으로 success가 나왔습니다.

pid에 현재의 프로세스값이 저장되므로 자기 자신에게 시그널을 보내도록 구현해주면 됩니다.

sigint\_handler와 sigstp\_handler에 이미 구현이 되어 있습니다.