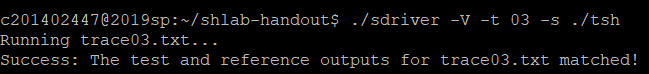
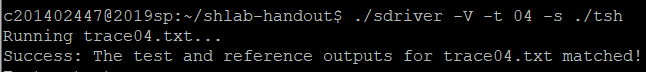
**2019 시스템 프로그래밍**

**- week07 -**

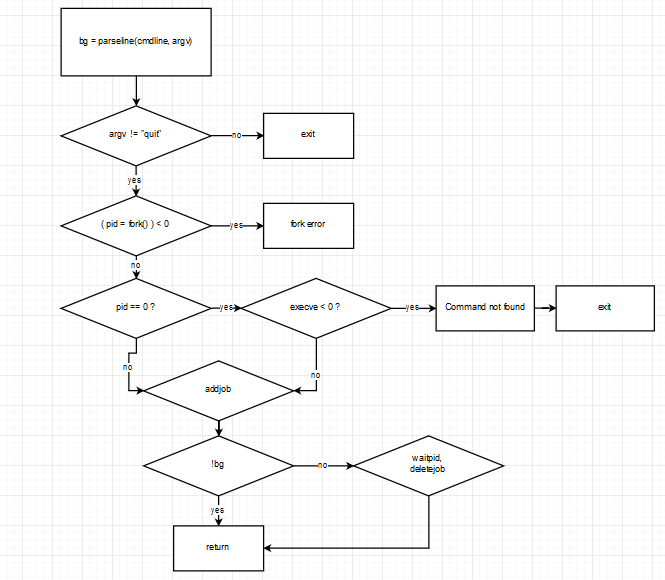
|  |  |
| --- | --- |
| **제출일자** | 2019.11.20 |
| **분 반** | 02 |
| **이 름** | 한원희 |
| **학 번** | 201402447 |

|  |
| --- |
| **Trace 03, 04** |

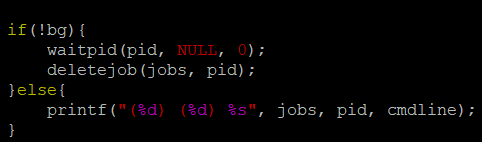




|  |
| --- |
| 03, 04 플로우 차트 |



|  |
| --- |
| Trace03, 04 해결 방법 설명 |



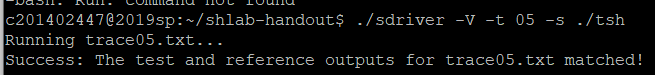
Foreground 작업의 shell과의 동기화(입력인자 x)와 입력인자가 있을때 foreground 작업의 동기화 입니다.

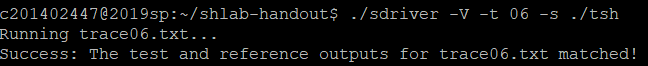
if 내부의 부분인데 !bg는 백그라운드가 아니라면(foreground이라면) if 내부로 들어갑니다.

waitpid는 부모 프로그램이 자식 프로그램을 기다리는 명령어 입니다.

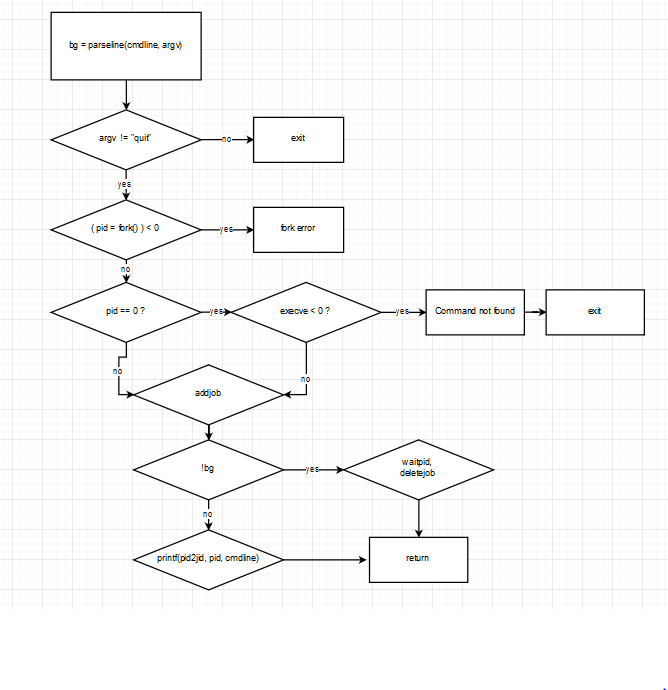
그 이후 등록된 joblist에서 삭제해줍니다. else 부분은 백그라운드에서 실행되는 job 부분입니다.

|  |
| --- |
| **Trace 05, 06** |

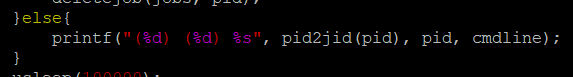




|  |
| --- |
| 05, 06 프로우 차트 |

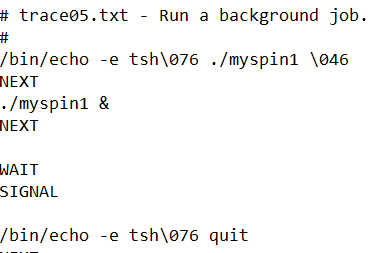


|  |
| --- |
| Trace 해결 방법 설명 |



위 3,4 의 해결방법에서 if문에 else문을 추가해줍니다. 즉 bg 부분인 백그라운드 입니다.

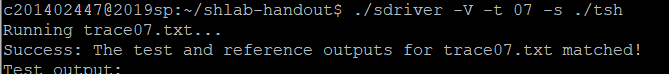
백그라운드일시에 printf를 실행해 줍니다.



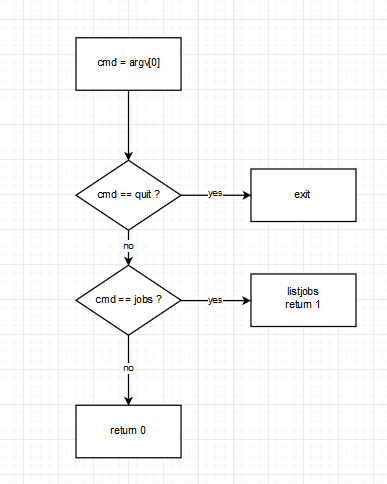
텍스트 파일을 확인해보면, job id, pid, 입력된 명령어가 출력되는 것을 확인할 수 있습니다.

pid2jid 명령어는 pid의 job id 를 불러와 주는 함수입니다.

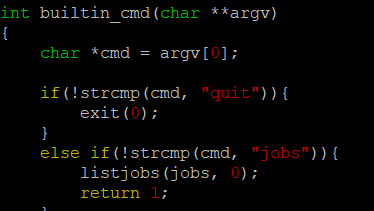
|  |
| --- |
| **Trace 07** |



|  |
| --- |
| 02 프로우 차트 |



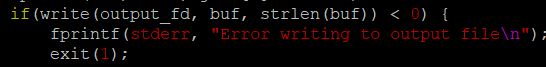
|  |
| --- |
| Trace 해결 방법 설명 |





builtin\_cmd에 새로운 명령어를 추가합니다. jobs를 치면 현재 작업 리스트를 쭉 보여줍니다.

구현하기 위해선 listjobs 함수를 사용해야 합니다. listjobs 함수를 본다면 인자를 두개 받습니다. jobs에는 jobs가 들어가고 output\_fd 항목을 본다면



이런식으로 write 함수 내부에서만 사용합니다. write함수가 0이하일때 에러 메시지를 출력하고 종료시킵니다. write 함수의 첫번째인자는 fd로 데이터 전송 영역을 나타내는 파일 디스크립터입니다. 파일 디스크립터는 0일때 input, 1일때 output, 2일때 error입니다. input을 주어야 하므로 listjobs의 두번째 인자는 0이 들어가야 합니다. 그리고 저때 return 0를 하면 오류가 납니다. 자세히는 모르겠지만 아마 현재 돌아가는 프로그램이기 때문에 return 1로 해주어야 하는것 같습니다.