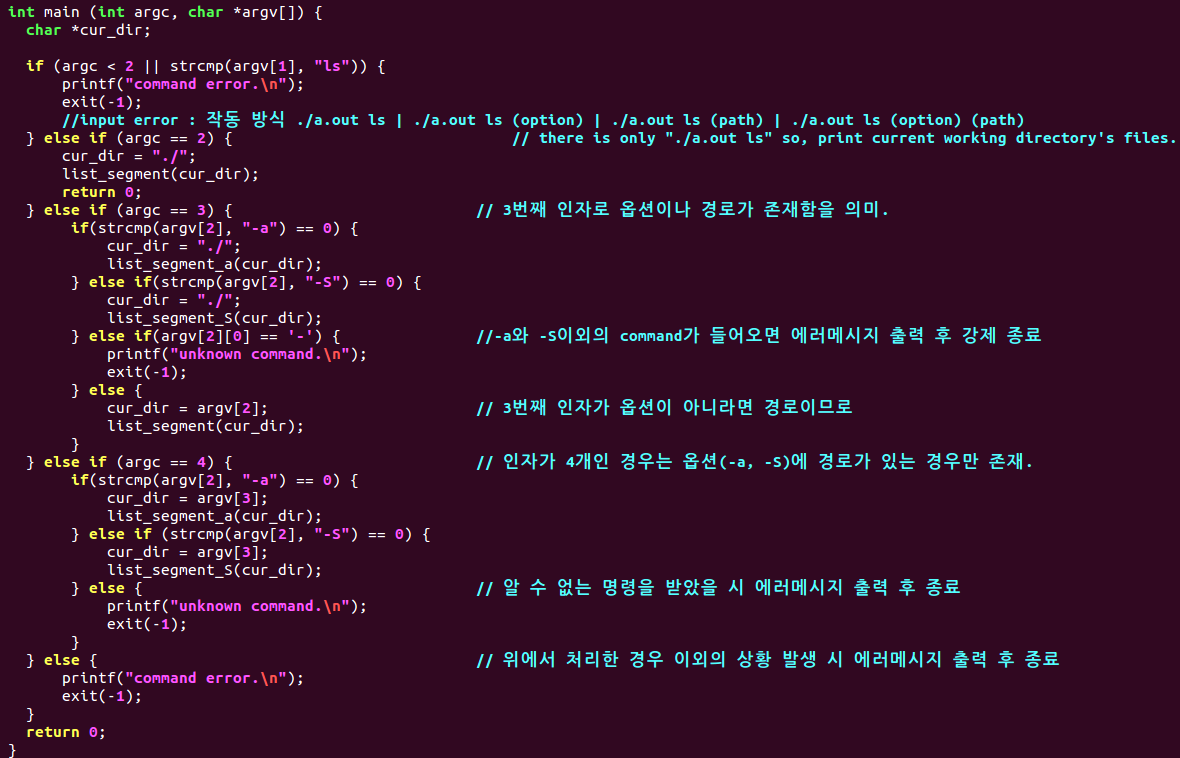
시스템 프로그래밍 보고서

**ls 커맨드 구현 (-S, -a)**

2012018109

컴퓨터공학부 컴퓨터전공

권현우

**1. main 함수** 

파일을 실행하면서 곧바로 main함수에 인자를 받도록 하였다.

파일의 실행방법은 “./a.out ls” 을 기본으로 한다.

입력되는 인자의 개수(argc)에 따라서 크게 2개 미만, 2개, 3개, 4개, 그 외의 경우로 구분했다.

단순히 “./a.out” 을 입력하는 경우, “./a.out” 뒤에 ls가 아닌 경우에는 command error 출력과 함께 프로그램이 종료된다.

입력된 명령을 실행한 이후에는 곧바로 프로그램을 종료하도록 하였다.

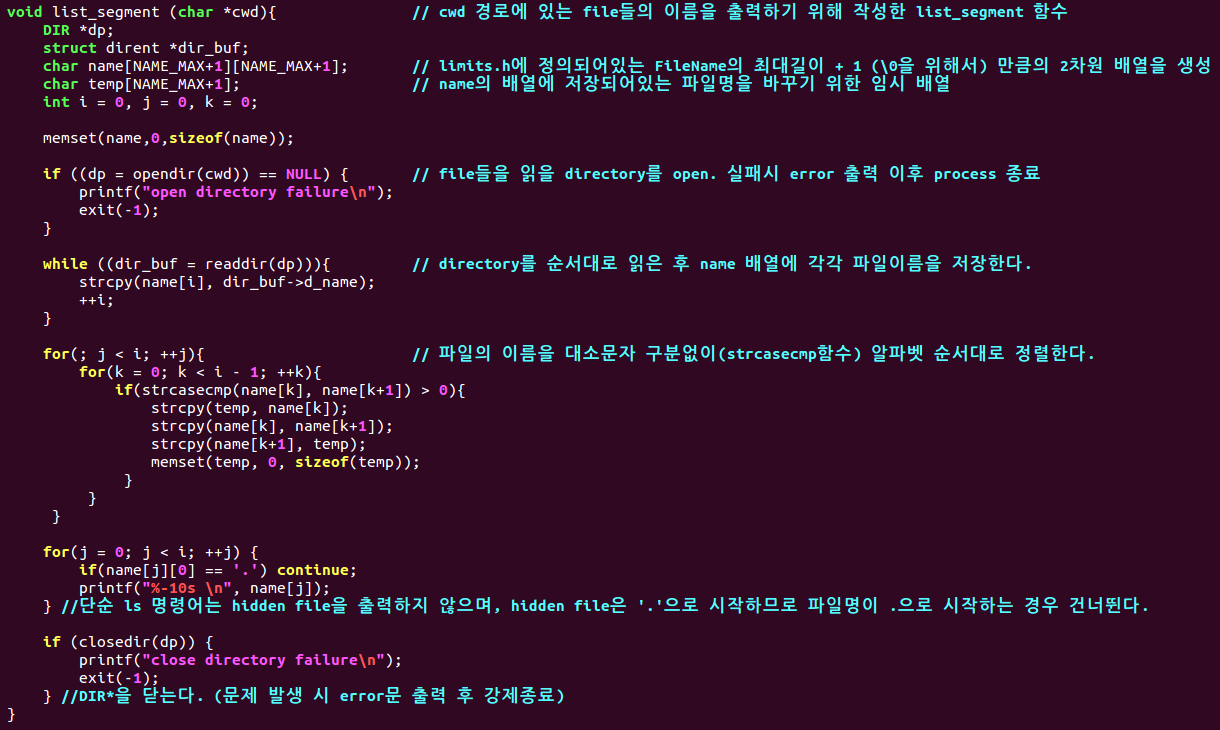
**2개의 경우** “./a.out ls” 밖에 존재하지 않으므로 cur\_dir를 “./”로 지정해준 뒤 list\_segment를 호출하여 현재 directory의 파일명을 출력하도록 한다.

**3개의 경우** “./a.out ls (option)” 와 “./a.out ls (path)” 2가지의 경우가 존재하므로 argv[2]가 option인지 path인지 구분하여 option인 경우에는 현재 경로(“./”)에 대해서 요구하는 옵션(-a/-S)에 맞게 파일이름을 출력하도록 하였으며, 만약 요구하는 옵션이

–a 혹은 –S가 아닌 경우에는 “command error”를 출력한 후 프로그램을 종료한다.

path의 경우 cur\_dir에 argv[2]를 대입하여 해당 경로에 대해 ls를 하도록 하였다.

**4개인 경우에는** “./a.out (option) (path)”의 경우밖에 존재하지 않으므로 입력한 path에 대해 요구한 option에 맞춰서 결과를 출력하도록 하였다. 만일 option이 잘못 되었을 경우 command error 출력 후 프로그램을 종료한다.

**2. list\_segment 함수 (ls) **

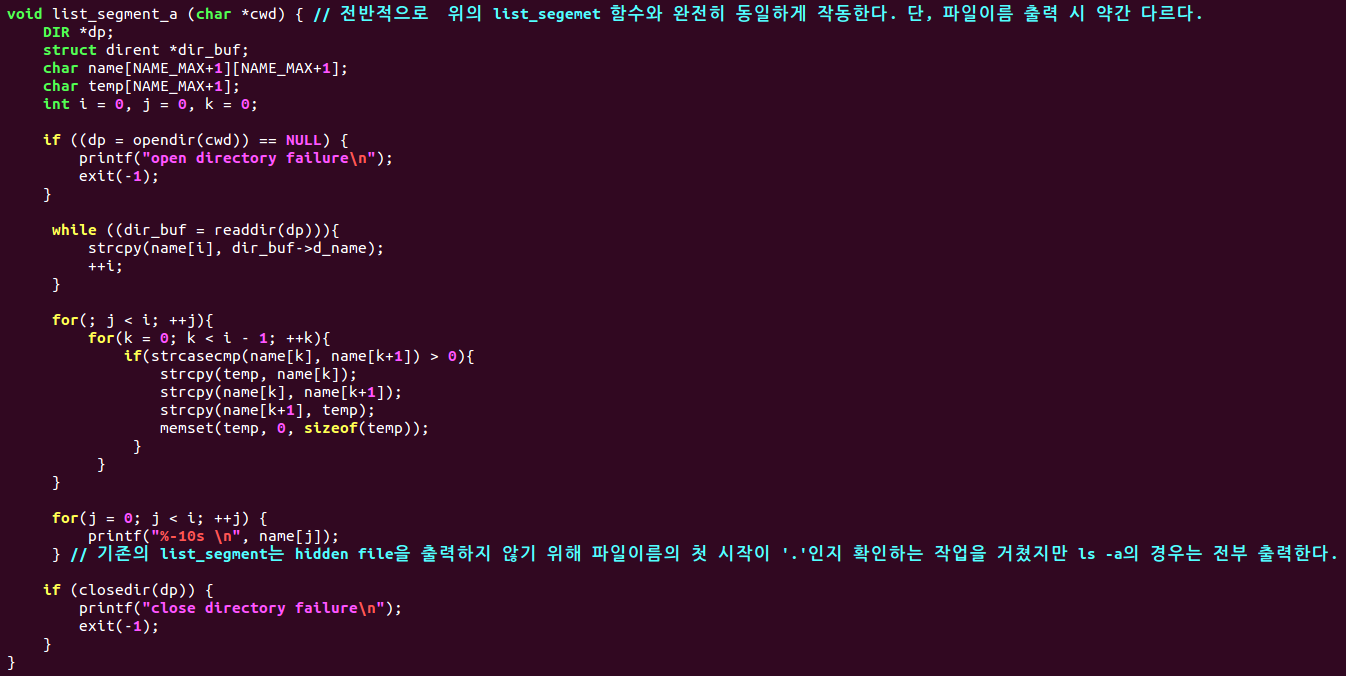
인자로 path를 받는다. (cwd)

받은 path를 통해 DIR\*를 열 수 없는 경우에는 “open directory failure”를 출력 후 종료한다.

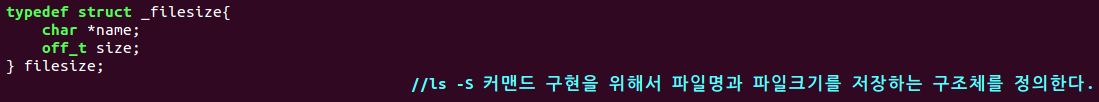
만약 아니라면, readdir을 통해서 파일을 순차적으로 읽어가면서 name에 해당 파일 이름을 저장한다. (NAME\_MAX를 통해 파일이름 최대 길이에 +1를 한다 (null문자 위해))

이후 파일의 이름을 대소문자 관계없이 알파벳의 순서로 정렬하기 위해서 strcasecmp를 사용하였다. 파일이름을 알파벳 순서로 정렬한 이후, 기본 ls의 경우 hidden file은 출력하지 않으므로 파일이름이 ‘.’로 시작하는지 확인 후 이에 해당하지 않는 경우에만 파일이름을 출력하도록 한다.

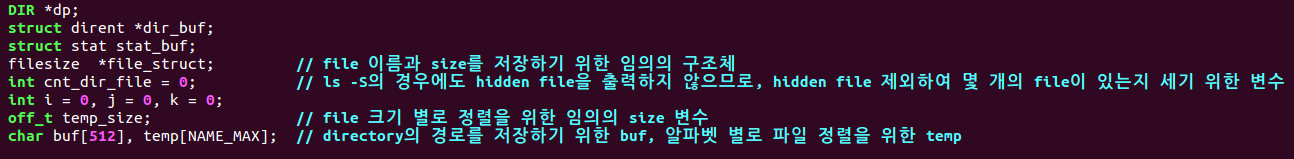
파일이름을 모두 출력한 이후 열어준 DIR\*을 닫아준다. 닫을 때도 오류 체크를 한다.

**2. list\_segment\_a 함수 (ls –a) **

기본적인 작동방식은 list\_segment와 완전히 동일하므로 차이점에 대해서만 설명하면 ls –a의 경우에는 숨김파일에 대해서도 모두 출력을 해야하므로 list\_segment에서 파일이름을 출력할 때 ‘.’로 시작하는가의 여부를 체크한 코드만 삭제해주었다.

**3. list\_segment\_S 함수 (ls –S) **

ls-S command의 경우 파일의 크기에 따라 크기가 큰 순서부터 차례대로 정렬을 하여 출력해야 하기 때문에 이를 위해 파일이름과 파일크기를 저장할 임의의 구조체 filesize를 정의하였다.

****

List\_segment\_S에서 사용하는 변수에 대한 기본적인 설명이다.

변수 “cnt\_dir\_file”은 hidden file을 제외하여 현재 directory 내에 몇 개의 파일이 존재하는지 확인하기 위한 변수이다.

변수 “temp\_size”는 파일의 크기를 비교하여 정렬하기 위한 임의의 변수이다.

변수 “buf”는 stat함수 호출 시 directory의 경로를 지정하기 위한 변수이다.

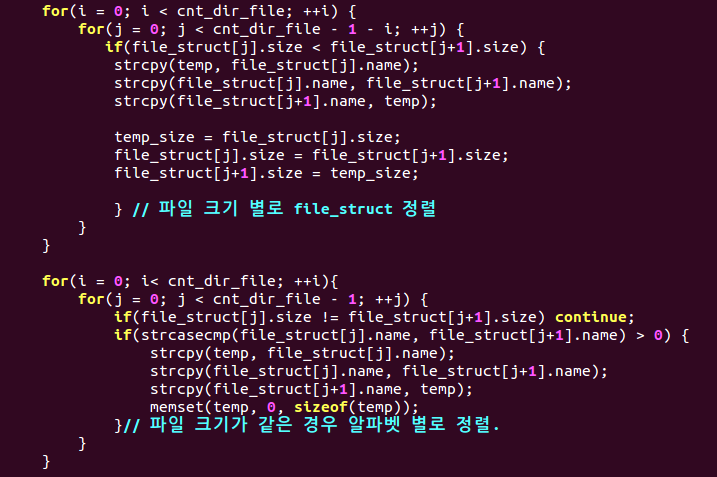
변수 “temp”는 알파벳 별로 파일 정렬을 위한 변수이다.



현재 directory 내에 몇 개의 파일(hidden file 제외)이 존재하는지 확인하기 위해 readdir을 통해 하나씩 읽어가면서 cnt\_dir\_file의 값을 1 증가시킨다.

이후 dp변수가 다시 첫 번째 파일을 가리키도록 rewinddir함수를 사용한다.

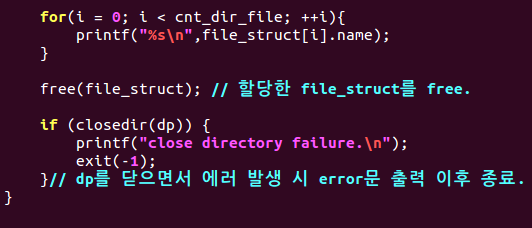
file\_struct를 cnt\_dir\_file의 개수만큼 동적할당하며, 다시 readdir를 통해 directory 내의 파일을 읽어가면서 file\_struct에 파일이름과 파일크기를 저장한다.

****

file\_struct끼리 파일 크기를 비교하여 크기가 더 작은 경우 뒤로 내보낸다.

만약 파일이름이 바뀐 경우에는 파일 크기도 바꾸어준다.

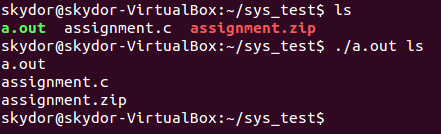
파일 크기에 따라 모든 파일을 정렬하면, 같은 크기의 파일에 대해서는 알파벳 순서로 정렬해준다.



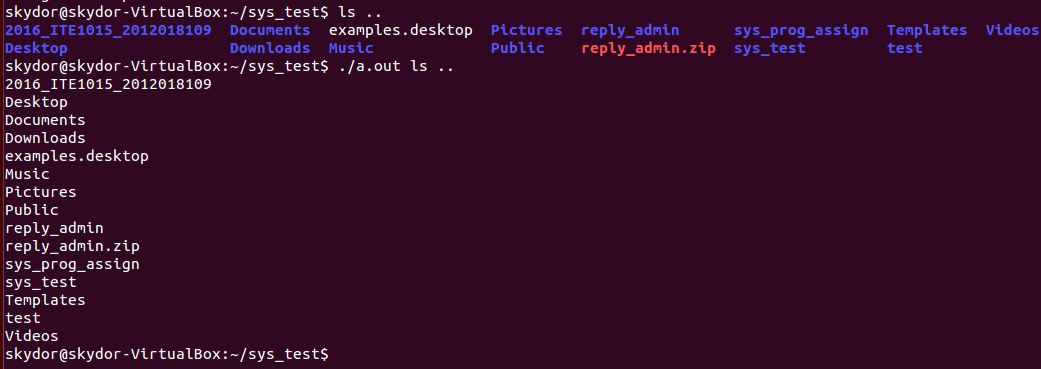
모든 파일이 정렬되면 파일 이름을 순서대로 출력하고 동적할당한 file\_struct를 free시킨 뒤 directory를 닫아준다.

**4. 실행 예시**

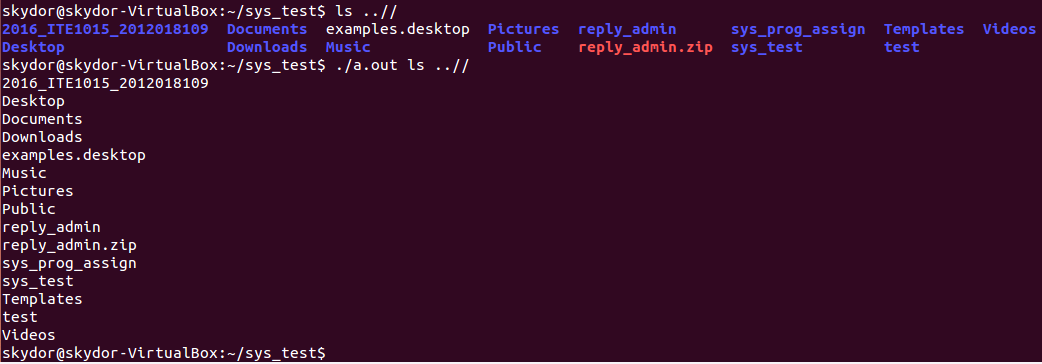
(1) 단순 ls 커맨드

****

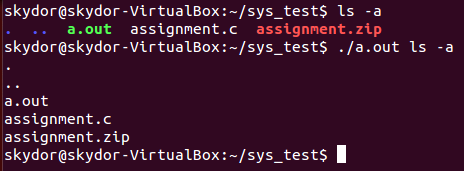
(2) ls (path) 커맨드



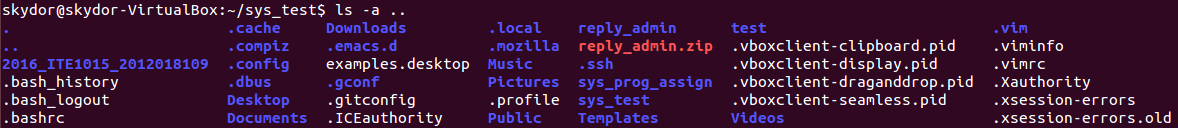
(3) ls (path) 커맨드 2 (경로 뒤에 /를 여러 개 넣어보았을 때)

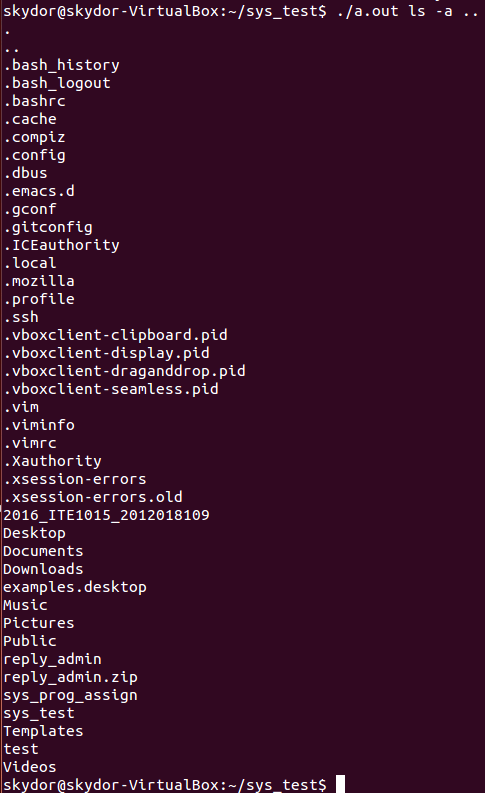


(4) ls –a 커맨드

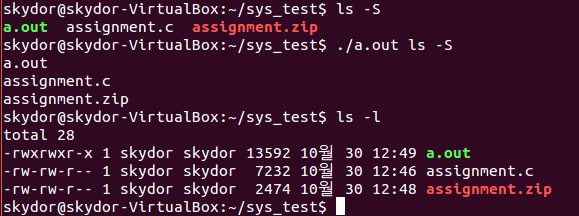


(5) ls –a (path) 커맨드

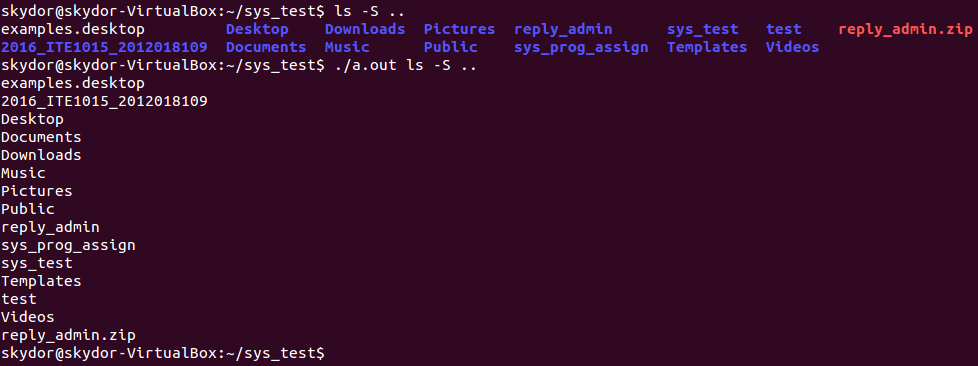


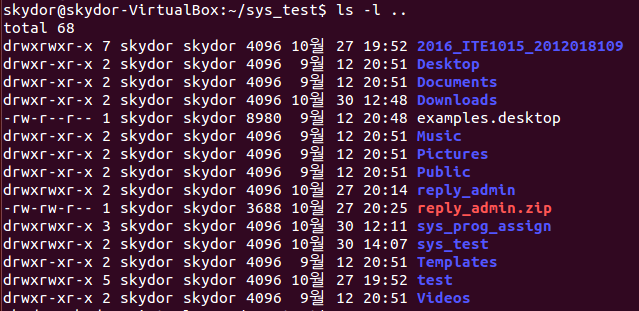


(6) ls –S 커맨드



(7) ls –S (path) 커맨드





제대로 구현되었는지 확인을 위해 파일 크기 체크

(8) 잘못된 커맨드 입력 예시

