컴퓨터학부 20192403 박상철

**개요**

다양한 시그널을 통해 데몬 프로세스를 생성하는 법을 익히 디렉토리를 관리하는 법을 익히는 것을 목적으로 하는 모니터링 프로그램

**기능**

**메인 기능 : 데몬 프로세서가 생성되면 해당 디렉토리를 사용자가 제시한 시간 간격으로 모니터링 하면서 파일 및 디렉토리에 삭제, 추가, 수정 등이 있으면 변경 사랑 파일에 기록**

**add : 데몬 프로세스 생성**

**-t : 프로세스 모니터링 시간 간격 설정**

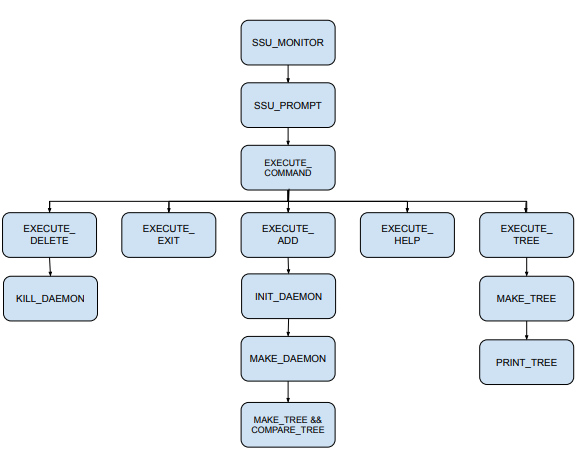
**delete : 데몬 프로세스 킬**

**tree : 디렉토리 구조 출력**

**exit : 프로그램 종료**

**help : 프로그램 사용법 출력**

**프로그램 설계도**



**상세 설계**

**프로그램이 실행되면 가장 먼저 suu\_mornitor함수에서 ssu\_promppt함수를 실행하여 실행할 명령어를 입력받고 excute\_command함수에서 입력받은 명령어에 해당하는 함수를(execute\_add, execute\_delete, execute\_tree, execute\_help, execute\_exit) 호출한다.**

**execute\_add가 실행되면 인자로 들어온 경로를 절대 경로로 바꾸고 예외처리(올바르지 않은 인자 혹은 경로 등) 과정을 거친뒤 init\_daemon함수를 호출한다. init\_daemon 자식 프로세스를 생성한 후 make\_daemon함수를 통해 데몬 프로세스를 만들고 monitor\_list파일에 추가한다, make\_daemon 프로세스는 데몬 프로세스의 모든 입출력을 막고 signal신호를 무시하게 만든다. 생성된 데몬 프로세스는 사용자가 입력한(입력하지 않았다면 1초) 시간 간격으로 make\_tree 함수로 트리를 만들고 compare\_tree 함수로 파일의 수정, 삭제, 추가 등의 사항을 모니터링 하여 기록한다.**

**execute\_delete는 monitor\_list에서 인자로 들어온 pid가 있는지를 확인하고 있다면 kill\_daemon함수를 호출하여 데몬 프로세스를 종료하고 파일에서 프로세스 정보를 삭제한다.**

**execute\_tree 함수는 인자로 들어온 경로가 파일에 있는지를 판별하여 있다면 모니터링 중인 디렉토리를 print\_tree 함수로 트리 형태로 출력한다.**

**execute\_help함수는 프로그램의 사용법(Usage)를 출력한다.**

**execute는 실행 프로그램을 정상 종료한다.**

void ssu\_monitor(int argc, char \*argv[]);

void ssu\_prompt(void);

int execute\_command(int argc, char \*argv[]);

void execute\_add(int argc, char \*argv[]);

int check\_add\_option(int argc, char \*argv[]);

int except\_add(char \*path);

void execute\_delete(int argc, char \*argv[]);

void kill\_daemon(pid\_t pid, char \*dirpath);

void execute\_tree(int argc, char \*argv[]);

void execute\_help(int argc, char \*argv[]);

void execute\_exit(int argc, char \*argv[]);

void init\_daemon(char \*dirpath, time\_t mn\_time);

pid\_t make\_daemon(void);

tree \*create\_node(char \*path, char \*filename, mode\_t mode, time\_t mtime);

void make\_tree(tree \*dir, char \*path);

void compare\_tree(tree \*old, tree \*new, FILE \*fp);

void print\_tree(tree \*node, int depth);

void free\_tree(tree \*cur);

void signal\_handler(int signum);

int scandir\_filter(const struct dirent \*file);

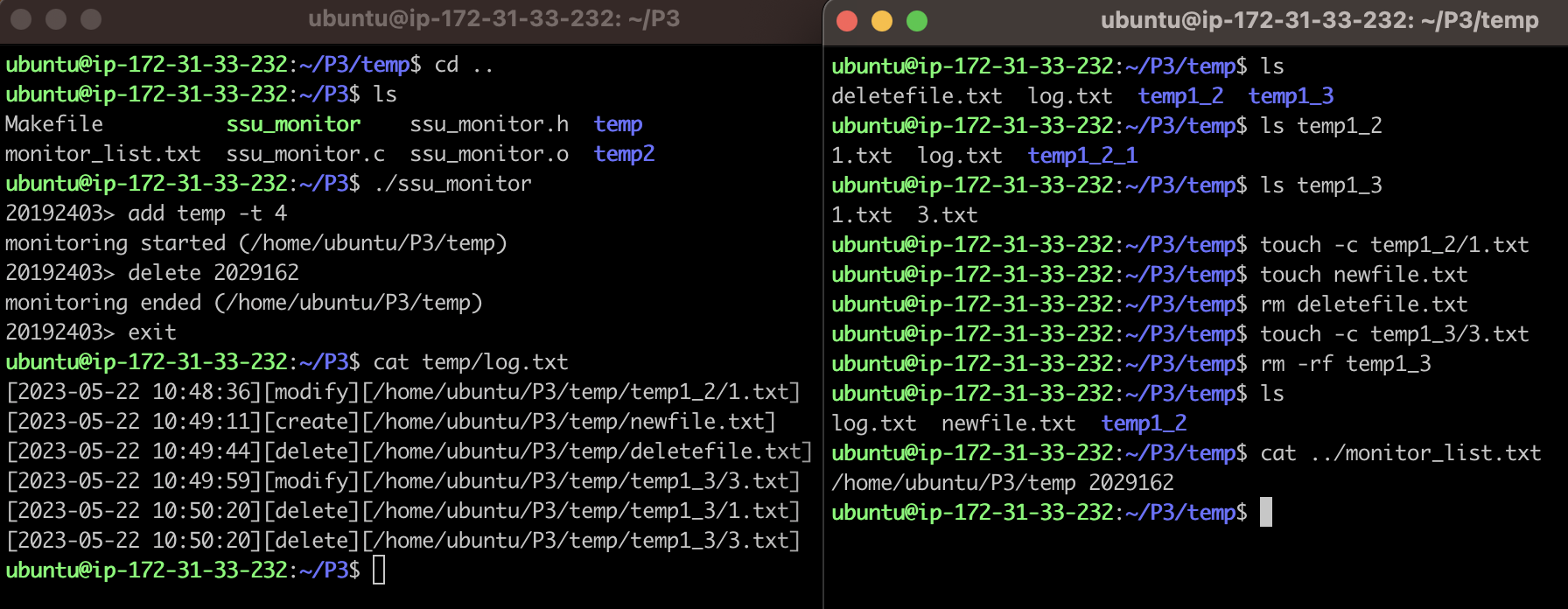
int splitByCh(char dest[][FILELEN], char \*str, char \*ch);

int get\_filetype(char \*path);

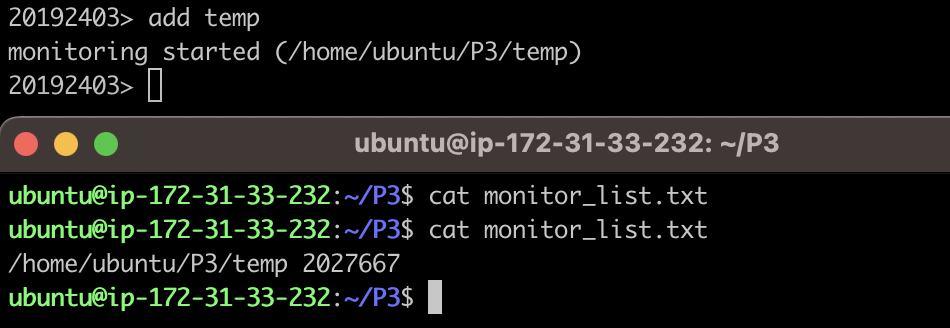
void printFormattedTime(time\_t mtime, FILE \*fp);

**실행 결과**

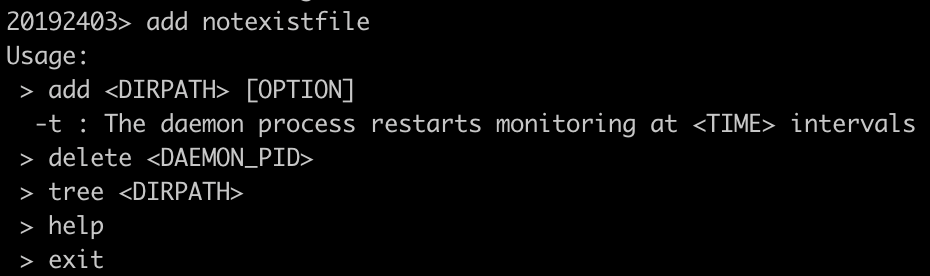
**모니터링 실행 장면**



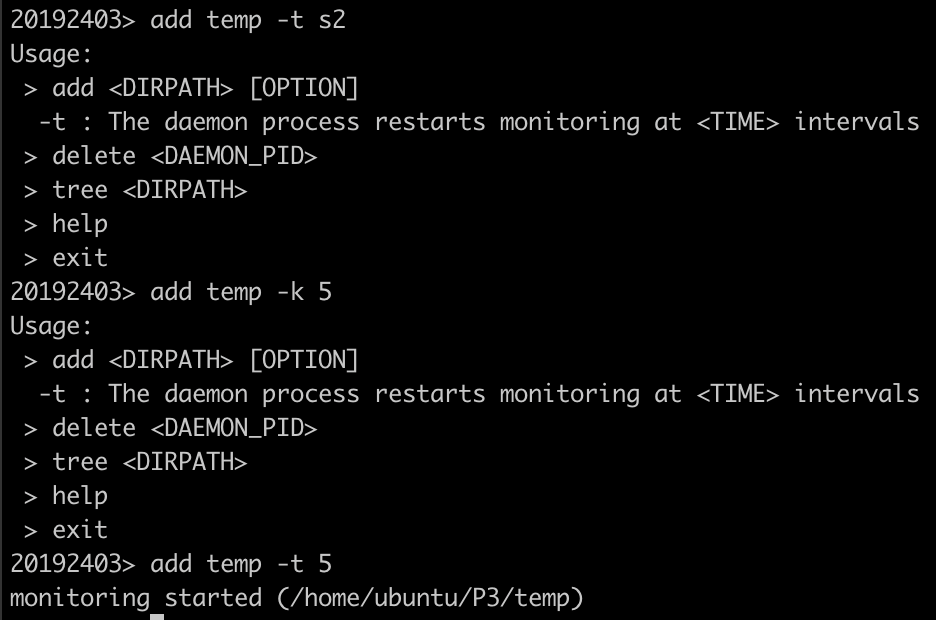
**add 실행화면**



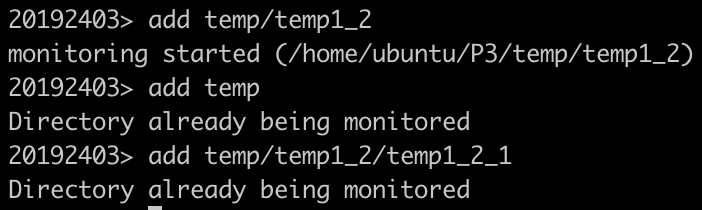
**add 예외 (존재하지 않는 파일)**



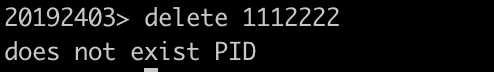
**add 예외 (잘 못된 t 옵션)**



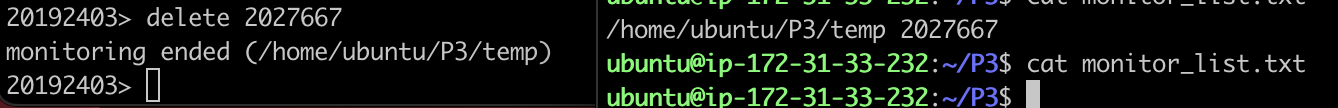
**add 예외 (이미 모니터링 중인 디렉토리 포함 했을 때)**



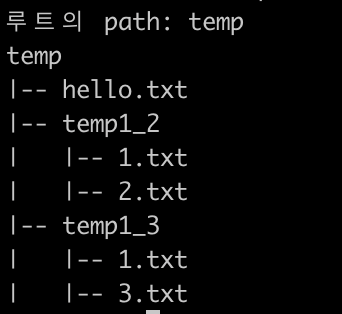
**delete 실행파일**



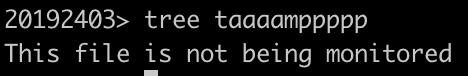
**delete 예외**



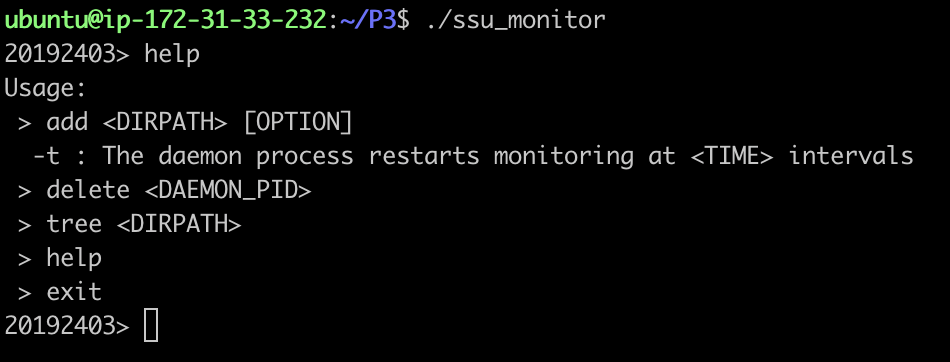
**tree 실행파일**



**tree 예외**



**help 실행**



**exit**



**prompt 예외**

