컴퓨터학부, 20190511, 배준형

# 과제 개요

|  |
| --- |
| ssu\_monitoring 프로그램은 디몬 프로세스를 이용하여 백그라운드로 프로세스가 실행되면서 디렉토리 폴더를 지정하면 해당 디렉토리내부의 파일들이 수정, 삭제, 생성 되었는지 확인하는 프로그램이다. 예를들어 add temp 라고 입력하면 현재경로가 /home/junhyeong 라고 헀을 떄 /home/junhyeong/temp 폴더를 모니터링하여 해당 폴더 내의 a.txt를 생성하면 [시간][create][/home/junhyeong/temp/a.txt] 라고 temp 폴더 내의 log.txt 파일에 기록이 될 것이다.  지원하는 옵션으론 {add, delete, tree, exit, help} 명령어가 존재한다.   * add <DIRPATH> * add <DIRPATH> -t <TIME> * delete <DAEMON\_PID> * tree <DIRPATH> * help * exit |

# 구현기능

## 실질적 구현 내용

|  |
| --- |
|  |

## ssu\_monitor 프로세스 프로토타입

|  |
| --- |
| [**프로토타입 및 함수 구현기능]**  typedef struct pn { //상대경로-->절대경로 이중포인터 구조체      char path[MAXPATHLEN];      struct pn \*prev;      struct pn \*next;  }pathNode;    typedef struct mt { //ssu\_montitor 파일 이중연결리스트      char path [MAXPATHLEN];      char pid [PIDSIZE];      struct mt\* next;      struct mt\* prev;  }Mnode;  typedef struct mtl{ //Mnode 연결리스트 관리 구조체      Mnode \*head;      Mnode \*tail;      int cnt;  }Mlist;  char\*\* path\_arr(char\* str); // str을 / 기준으로 토큰배열화  int realpathS (char \*str); // 링크드리스트기반 경로찾기 함수  char\*\* prom\_args (char\* str); // 프롬포트 토큰 배열 생성  char\* tokinzer (char\* str, char\* del); // del tokenizer  int path\_setting ();   // 현재경로, 디몬경로, log.txt 파일 경로를 알려주는 함수  int print\_tree (char\* path, int depth);  Mnode\* newM (char\* path, char\* pid);  Mlist\* newMlist ();  Mnode\* appendM (Mlist\* list, char\* path, char\* pid);  int delNode (Mlist\* list, char\* path);  // return 은 해당 pid  int free\_mlist (Mlist\* list); //list를 요소를 모두 free 하는 함수.  void printM (Mlist \*list); // list의 모든 요소를 출력해주는 함수.  Mlist\* read\_monitorfile(); //리스트의 있는 데이터를 moitor\_list.txt 로부터 입력  int isExistMlist (Mlist\* list, char\* path); // 해당 리스트에 path가 있는지 확인  int canAddMlist (Mlist\* list, char\* path); // 해당 리스트에 path가 같거나 포함되거나, 포함하거나 하면 오류처리  int write\_monitorfile(Mlist\* list); //리스트의 있는 데이터를 moitor\_list.txt 에 출력  int open\_monitor\_read();    //monitor\_list.txt read 오픈 전용  int open\_monitor\_write();   //monitor\_list.txt write 오픈 전용  void print\_help(); //help 출력  int directory\_check (char\* path); //딕셔너리가 실행중인 디몬 프로세스 경로와 겹치는지 확인  int do\_add(int argc, char\* args[]); // add옵션 수행  int do\_delete(int argc, char\* args[]); // delete 옵션 수행  int do\_help(int argc, char\* args[]); // help 옵션 수행  int do\_tree(int argc, char\* args[]); // tree 옵션수행  int do\_module(int option, int argc, char\* args[]); // add,delete,help,tree 옵션 수행 |

## daemon.c 프로세스 프로토타입

|  |
| --- |
| [**프로토타입 및 함수 구현기능]**  //디몬 프로세스 구조체 (이중연결리스트 형태)  typedef struct lp {      char path [MAXPATHLEN];        time\_t mtime; // 최종 수정 시간      int visit;    // 생성 시 1, 디몬에서 monitoring 할 때 0으로 변경-->  다시확인할 때 0인지 확인!        struct lp\* prev;      struct lp\* next;  }LogPath;  typedef struct lpl {      LogPath \*head;      LogPath \*tail;      int cnt;  }LogpathList;    //\*\* 디몬 프로세스꺼  int make\_time (time\_t\* t, char\* buf); // 시간 출력  LogPath\* newL (char\* path);  LogpathList\* newList (); // list 생성자  /\* path (디렉토리만 지원) 경로의 파일 모두 재귀호출하여 Regular File에 대해서 삽입\*/  LogpathList\* scandir\_list (LogpathList\* list\_arg, char\* path, int depth); // list\_arg 리스트 NULL 이면 새로 생성해서 return해줌  LogPath\* appendL (LogpathList \*list, char\* path);   // fl  LogPath\* pop\_path (LogpathList \*list, char\* path);  // list에서 path에 해당하는 노드 삭제  LogPath\* pop\_node (LogpathList \*list, LogPath\* node); // list에서 node 즉시 삭제  int compare\_list (LogpathList \*list, char\* path);   // 연결리스트 list 와 path를 비교하며 순회  int compare\_node (LogpathList \*list, char\* path); //list 의 있는 모든 노드와 path를 비교해주는 함수.  int check\_listvisit (LogpathList \*list);    // 리스트를 순회하며 1인지 0인지 체킹  int print\_logchar (LogPath\* node, int option, char\* print\_buf); // [시간][모드][경로] 를 print\_buf에 만들어 주는 함수  int final\_module (LogpathList \*list, char\* path, unsigned int timer); // daemon 프로세스에서 실행되는 loop 모듈  int printfile (int fd, char\* print\_buf); //fd에 파일 출력 함수  void printL (LogpathList \*list); // list의 모든 요소를 출력해주는 함수.  int exclusive\_swpfile (char\* path); //.swp 파일은 monitoring에서 제외하는 함수.  int log\_fd; //log file descriptor 전역변수  unsigned int time\_sleep = 1; // -t <TIME> 옵션  int daemon\_init(void);  void handle\_sigusr1(int signo);  char daemon\_path [MAXPATHLEN];  char monitor\_path [MAXPATHLEN];  char target\_path [MAXPATHLEN];  char log\_path[MAXPATHLEN+10]; //log 경로 |

# 상세설계

## ssu\_monitor.c

|  |
| --- |
| 1. **[모듈 구성]** 2. 우선 ssu\_montior 프로세스에서 add <DIRNAME> 를 입력하면 fork->exec를 통해 디몬 프로세스를 출력한다. |

# 실행결과

## add 옵션 // add t 옵션

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 웹사이트이(가) 표시된 사진

자동 생성된 설명

## delete 옵션

* 텍스트, 스크린샷, 폰트이(가) 표시된 사진

  자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

## tree 옵션

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

## help 옵션

텍스트, 스크린샷, 폰트, 직사각형이(가) 표시된 사진

자동 생성된 설명

## exit 옵션

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

## makefile

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명