컴퓨터학부 20201841 박세연 llilil

1. 과제개요

ssu\_backup 프로그램은 사용자가 어떤 원본 파일이나 디렉터리에 대한 백업 파일을 생성, 삭제, 복구하는 기능을 제공하는 프로그램이다. 주요 명령어로는 backup, remove, recover, list, help가 있다.

backup은 사용자에게 원본의 경로를 입력 받으면 그에 해당하는 백업 파일이 /home/backup 디렉터리 아래에 일자 디렉터리를 생성하고 일자 디렉터리 아래에 백업 파일을 생성한다. 생성하려는 백업 파일이 이전에 생성한 백업 파일과 같은 내용이라면 중복 처리를 통해 이미 존재하는 백업 파일임을 알리고 백업을 진행하지 않는다. remove는 원본의 경로를 입력 받아 그에 해당하는 백업 파일을 삭제하는 명령어이다. 만약 해당 원본 경로에 있는 백업파일이 여러 개 존재한다면 사용자에게 파일의 번호를 입력 받아 해당 파일을 삭제할 수 있다.

recover는 사용자에게 원본 파일의 경로를 입력 받아, 해당 경로와 일치하는 백업 파일을 복구하는 명령어이다. 복구하려는 백업 파일과 현재 원본 파일의 내용이 같은 경우에는 내용이 같음을 알리고 복원을 진행하지 않는다. remove 명령어와 마찬가지로 그에 해당하는 백업 파일이 여러 개 존재한다면 사용자에게 파일의 번호를 입력 받아 해당 백업 파일로의 복구를 진행한다.

list 명령어는 사용자에게 원본의 경로를 입력 받아 해당 경로의 백업 디렉터리 내의 파일 및 디렉터리를 트리 형태로 출력하고 rm, rc, vi(m), exit의 사용자 입력에 대해 명령을 수행하는 기능을 제공한다. rm은 remove와 같은 역할을 하고, rc는 recover와 같은 역할을 한다. vi(m)은 실제 리눅스 명령어인 vi(m)과 같은 기능을 제공한다. exit를 입력하기 전까지는 계속해서 사용자에게 입력을 받을 수 있으며 exit 명령 입력 시 프로그램이 종료된다.

help는 위에서 설명한 내장 명령어에 대한 설명을 출력해주는 명령어이다. help 뒤에 명령어 인자를 넘겨주지 않으면 전체 명령어에 대한 usage를 볼 수 있으며, 첫번째 인자로 특정 명령어를 입력한다면 해당 명령어의 usage만 단독으로 볼 수 있다.

주요 명령어 중에서도 backup, remove, recover 명령어는 각 명령어에서 사용할 수 있는 추가적인 옵션이 존재하며 사용자의 요구에 맞는 옵션을 받아 처리할 수 있다.

2. 기능

**<ssu\_header.h>**

char exeNAME[PATHMAX]; // 실행 파일의 이름(ssu\_backup)을 저장할 변수

char exePATH[PATHMAX]; // 실행 경로를 저장할 변수

char homePATH[PATHMAX]; // /home/사용자이름

char backupPATH[PATHMAX]; // 백업 디렉터리 경로 (/home/backup)

char logfilePATH[PATHMAX]; // 백업 로그파일 경로 (/home/backup/ssubak.log)

int hash = HASH\_MD5; // md5의 해시 크기(33)를 담은 변수

char \*commanddata[10]; // 프로그램에 사용할 명령어를 모두 담고 있는 배열

// 사용자에게 입력받은 명령어와 파라미터들을 담는 구조체

typedef struct command\_parameter

{

char \*command; // 명령어

char \*filename; // 원본 파일 경로

char \*tmpname;

int commandopt; // 입력한 옵션의 정보를 저장

char \*argv[10]; // 옵션 뒤의 인자 등을 저장할 공간

} command\_parameter;

// 해시 값을 구할 경로를 target\_path로 넘겨, 그에 대한 md5 해시 값을 hash\_result에 저장하는 함수

int md5(char \*target\_path, char \*hash\_result);

// target\_path로 해시 값을 구할 경로를 넘겨, hash\_result에 저장하는 함수

int ConvertHash(char \*target\_path, char \*hash\_result);

// 로그파일을 읽어서 백업 파일의 연결리스트를 구축할 노드 구조체

struct LogBackupNode

{

char timestamp[15]; // 백업 시간 정보

char original\_path[4096]; // 백업되었거나 삭제된 원본 파일 경로

char backup\_path[4096]; // 백업된 파일의 경로

char action[256];

struct LogBackupNode \*prev;

struct LogBackupNode \*next; // 다음 노드를 가리키는 포인터

};

// 연결 리스트의 헤드 포인터

struct LogBackupNode \*logHead = NULL;

struct LogBackupNode \*logTail = NULL;

// list에서 출력할 때 사용하는 포인터

struct LogBackupNode \*mainHead = NULL;

struct LogBackupNode \*mainTail = NULL;

// list 명령어에서 입력받은 디렉터리에 대한 백업 파일들의 노드들을 추가할때 오름차순으로 추가하는 함수

void insertInOrder(struct LogBackupNode \*\*headRef, struct LogBackupNode \*newNode);

int countWords(char \*input); // 문자열을 '/'단위로 쪼개 몇개가 나오는지 반환하는 함수

char \*splitAndPrint(char \*str); // 문자열을 처음 나오는 / 로 쪼개 앞부분을 출력하고 뒷부분을 반환하는 함수

void deleteEmptyDirectories(const char \*basePath); // 빈 백업 디렉터리 삭제하는 함수

// 연결 리스트를 이용하여 중복된 파일을 확인하는 함수

// filepath 인자 -> 백업하려는 원본파일의 경로

// command 인자 -> backup에서 사용하면 0, recover에서 사용하면 1

int isDuplicateFile(char \*filepath, int command);

// 로그 백업 상태를 파일에서 읽어와 연결 리스트에 추가하거나 삭제하는 함수

// (+ backup 명령어인 경우에는 중복 검사) cmpPath : 중복검사하려는 원본 경로

// command로 명령어 구분 (0: backup, 1: remove, 2:recover, 3: list)

int readBackupLog(char \*logfilePath, char \*cmpPath, int command);

typedef struct backupNode backupNode; // 백업 파일들의 리스트를 형성하는 노드 구조체

typedef struct fileNode fileNode; // 특정 파일 정보를 저장할 노드

typedef struct dirNode dirNode ; // 디렉터리 정보를 저장할 노드

typedef struct dirList dirList; // 디렉터리 리스트를 관리할 구조체

typedef struct pathList\_ pathList; // 경로의 목록을 관리하는 구조체

void help(char \*command); // 명령어의 usage를 출력하는 help 명령어 함수

int cmpHash(char \*path1, char \*path2); // 해시값을 비교해주는 함수

char \*cvtNumComma(int a); // 정수를 입력받아 문자열로 변환, 세 자리마다 쉼표(,)를 추가하여 반환하는 함수

char \*getDate(); // 현재 일시를 특정한 형식의 문자열로 만들어 반환하는 함수

char \*QuoteCheck(char \*\*str, char del); // str에서 del을 찾아 그 위치 바로 앞의 문자를 제거하는 함수

char \*Tokenize(char \*str, char \*del); // str을 del(구분자)에 따라 토큰화 하는 함수

// str을 del 구분자로 토큰화하고, 그 결과 얻어진 토큰들의 배열을 반환하는 함수

// 분리된 토큰의 개수는 cnt 포인터를 통해 반환

char \*\*GetSubstring(char \*str, int \*cnt, char \*del);

int ConvertPath(char \*origin, char \*resolved); // origin을 표준화된 절대 경로로 변환해 resolved에 저장하는 함수

int cmpPath(char \*path1, char \*path2); // 두 경로가 일치하는지 판단하는 함수

**<ssu\_backup.c>**

// list 명령어를 실행 후, 사용자 명령을 받을 때 주어진 명령어를 사용하여 외부 프로그램을 실행시키는 함수

void SystemExec(char \*\*arglist);

// list 명령어를 실행 후, 사용자 명령을 받을 때 해당 명령어를 처리하는 인터페이스 역할

int Prompt(char \*path, char \*\*backupFileList, int backupFileCnt);

// list의 명령어 인자로 파일이 들어온 경우를 처리하는 함수

int ListFile(char \*path);

// list의 명령어 인자로 디렉터리가 들어온 경우를 처리하는 함수

int ListDir(char \*basePath);

// list를 수행하기 전 사용자의 명령 조건에 알맞게 처리하기 위해 실행하는 함수

int ListCommand(command\_parameter \*parameter);

// 파일을 복구하는 함수

int RecoverFile(char \*path, char \*newPath, int optDR, int optL);

// 디렉터리를 복구하는 함수

int RecoverDir(char \*dirPath, char \*newPath, int optDR, int optL);

// recover를 수행하기 전 사용자의 명령 조건에 알맞게 처리하기 위해 실행하는 함수

int RecoverCommand(command\_parameter \*parameter);

// 백업 파일을 삭제하는 함수

int RemoveFile(char \*path, int optA, int optL, int optDR);

// 백업 디렉터리를 삭제하는 함수

int RemoveDir(char \*dirPath, int optDR, int optA, int optL);

// remove를 수행하기 전 사용자의 명령 조건에 알맞게 처리하기 위해 실행하는 함수

int RemoveCommand(command\_parameter \*parameter);

// 파일을 백업하는 함수

int BackupFile(char \*path, char \*date, char \*basedir, int optY);

// 디렉터리를 백업하는 함수

int BackupDir(char \*path, char \*date, char \*basedir, int optY);

// backup을 수행하기 전 사용자의 명령 조건에 알맞게 처리하기 위해 실행하는 함수

int AddCommand(command\_parameter \*parameter);

// 인자 배열 arglist를 기반으로 특정 명령을 실행하는 함수를 호출하는 역할

void CommandFun(char \*\*arglist);

// parameter 구조체로 명령을 실행하는 프로세스를 생성하고 실행하는 함수

void CommandExec(command\_parameter parameter);

// parameter 구조체 멤버 초기화

void ParameterInit(command\_parameter \*parameter);

// 명령어에 따라 인자를 처리하고 처리 결과를 parameter 구조체에 저장하는 함수

int ParameterProcessing(int argcnt, char \*\*arglist, int command, command\_parameter \*parameter);

// 함수 실행을 위한 초기화 함수

void Init();

**<ssu\_help.c>**

// help 명령어를 실행하는 함수로 command 인자로 출력하려는 명령어 받음

// 사용자가 인자를 생략할 경우 임의로 “all” 문자열을 command로 받아 모든 명령어에 대한 usage 출력

void help(char \*command);

void helpBackup(); // backup 명령어에 대한 usage를 출력하는 함수

void helpRemove(); // remove 명령어에 대한 usage를 출력하는 함수

void helpRecover(); // recover 명령어에 대한 usage를 출력하는 함수

void helpList(); // list 명령어에 대한 usage를 출력하는 함수

void helpHelp(); // help 명령어에 대한 usage를 출력하는 함수

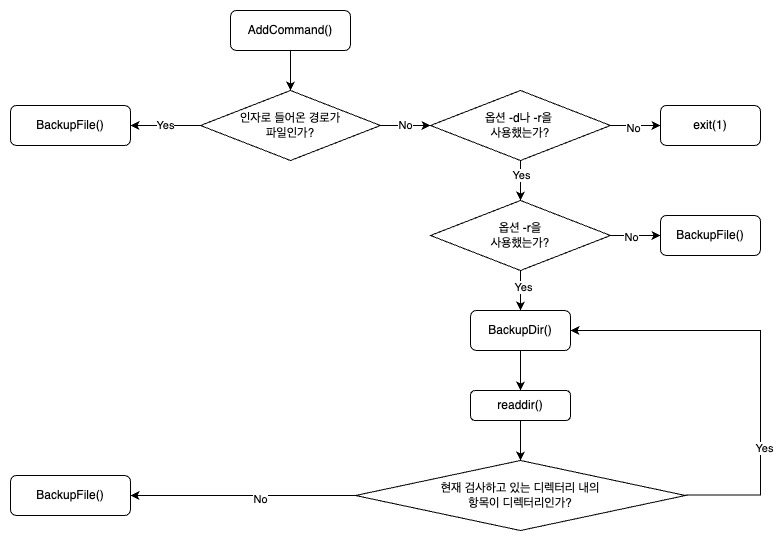
3. 상세설계

**<메인 흐름도>**

도표, 스케치, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

**<backup 명령어 – AddCommand() 흐름도>**



**<remove 명령어 – RemoveCommand() 흐름도>**

**도표, 종이접기, 라인, 스케치이(가) 표시된 사진

자동 생성된 설명**

**<recover 명령어 – RecoverCommand() 흐름도>**

도표, 종이접기, 라인, 스케치이(가) 표시된 사진

자동 생성된 설명

**<list 명령어 – ListCommand() 흐름도>**

도표, 라인, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

4. 실행결과

**(1) ssu\_backup**

텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

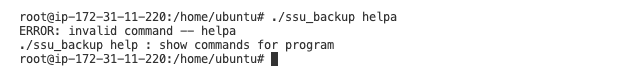
자동 생성된 설명

* help를 통해 명령어 목록을 출력한 모습

텍스트, 폰트, 화이트, 대수학이(가) 표시된 사진

자동 생성된 설명

* 첫번째 인자 입력하지 않을 시 에러처리 후 프로그램 종료



* 첫번째 인자로 잘못된 명령어를 입력할 경우

**(2) 내장 명령어 backup**

**텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명**

* backup 내장 명령어 실행

텍스트, 폰트, 스크린샷, 대수학이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷, 대수학이(가) 표시된 사진

자동 생성된 설명

* backup 명령어 예외처리

텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -d 옵션 입력시 <PATH>가 디렉터리일 시 해당 경로 아래에 있는 모든 파일들에 대한 백업

텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -r 옵션 입력 시 <PATH>가 디렉터리일 시 해당 경로 아래에 있는 모든 파일과 서브 디렉터리 내의 파일들에 대해서도 백업을 진행

텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -y 옵션 입력 시 <PATH>에 대한 경로와 내용이 같은 백업 파일이 존재해도 백업 진행

**(3) 내장 명령어 remove**

텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* remove 내장 명령어 실행

텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* 사용자 홈 디렉토리엔 없지만 버전 디렉터리에 있는 파일일 경우 (지워진 파일인 경우)

텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -d 옵션 입력 시 <PATH>가 디렉터리일 시 해당 경로 아래에 있는 모든 파일들의 백업 파일에 대해 삭제를 진행하고, 백업 파일이 2개 이상인 파일들은 목록을 출력하여 사용자 입력을 받은 후에 삭제

텍스트, 소프트웨어, 컴퓨터, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -r 옵션 입력 시 <PATH>가 디렉터리일 시 해당 경로 아래에 있는 모든 파일들의 백업 파일과 서브 디렉터리 내에 있는 모든 파일에 대해서도 삭제를 진행하고, 백업 파일이 2개 이상인 파일들은 목록을 출력하여 사용자 입력을 받은 후에 삭제

텍스트, 소프트웨어, 컴퓨터, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -r 옵션과 -d 옵션을 동시에 입력 하면 -r 옵션을 적용시켜 재귀적으로 백업 파일에 대한 삭제 진행

텍스트, 소프트웨어, 컴퓨터, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -a 옵션 입력시에는 백업본이 2개 이상인 파일들에 대해 목록을 출력하지 않고 모든 백업 파일을 삭제

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

* remove 명령어 예외처리

**(4) 내장 명령어 recover**

텍스트, 소프트웨어, 웹 페이지, 웹사이트이(가) 표시된 사진

자동 생성된 설명

* recover 내장 명령어 실행

텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* 사용자 홈 디렉토리엔 없지만 버전 디렉터리에 있는 파일일 경우 (지워진 파일인 경우)

텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -d 옵션 입력 시 <PATH>가 디렉터리일 시 해당 경로 아래에 있는 모든 파일들의 백업 파일에 대해 복구를 진행하고, 백업 파일이 2개 이상인 파일들은 목록을 출력하여 사용자 입력을 받은 후에 복구

텍스트, 소프트웨어, 컴퓨터, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -r 옵션 입력 시 <PATH>가 디렉터리일 시 해당 경로 아래에 있는 모든 파일들의 백업 파일과 서브 디렉터리 내에 있는 모든 파일에 대해서도 복구를 진행하고, 백업 파일이 2개 이상인 파일들은 목록을 출력하여 사용자 입력을 받은 후에 복구

텍스트, 소프트웨어, 스크린샷, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* -l옵션 입력 시 백업 파일이 2개 이상인 파일들에 대해 리스트를 출력하지 않고 가장 최근에 백업한 백업 파일을 복구

텍스트, 소프트웨어, 컴퓨터, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* n 옵션 입력 시에는 해당 옵션 뒤에 <NEW\_PATH>를 필수로 입력받아, 백업 파일에 대해 새로운 경로를 지정하여 복구를 진행하고, 만약 해당 경로가 없을 경우 디렉터리를 생성하여 경로를 설정

텍스트, 폰트, 스크린샷, 문서이(가) 표시된 사진

자동 생성된 설명

* recover 명령어 예외처리

**(5) 내장 명령어 list**

**텍스트, 소프트웨어, 웹 페이지, 컴퓨터이(가) 표시된 사진

자동 생성된 설명**

* list의 인자로 파일을 넘겨주었을 때 최근 백업 시간을 가지고 있는 백업 파일 하나를 출력

텍스트, 소프트웨어, 컴퓨터, 컴퓨터 아이콘이(가) 표시된 사진

자동 생성된 설명

* 사용자 입력을 기다리며 명령 실행 가능. vi(m) 입력시 해당 원본 파일에 대해 수정하고, exit 입력 시 프로그램이 종료

**(6) 내장 명령어 help**

**텍스트, 폰트, 스크린샷, 종이이(가) 표시된 사진

자동 생성된 설명**

* 프로그램 내장 명령어에 대한 설명(Usage) 출력