컴퓨터학부 / 20192393 / 김현우

**1 . 과제개요**

프로그램에서 만든 내장명령어 add , remove, recover, ls, vi(m), help, exit 명령어를 이용한다 .

**add** : 백업을 원하는 파일을 입력 받아 백업 디렉토리에 추가한다.

-d 옵션일 때 백업을 원하는 디렉토리를 입력 받아 백업 디렉토리에 추가한다.

**remove** : 백업된 파일을 입력 받아 백업 디렉토리 내에 존재하는 파일을 삭제한다.

-a 옵션일 때 인자로 넣어준 파일을 모두 삭제하거나, 인자로 넣어준 디렉토리와 디렉토리내에 있는 모든 파일을 삭제한다.

-c 옵션일 때 backup 디렉토리내에 있는 모든 파일을 삭제한다.

**recover** : 백업된 파일을 입력 받아 백업 디렉토리 내에서 현재 작업 디렉토리에 다시 복구한다.

-d 옵션일 때 디렉토리 내에 존재하는 모든 파일을 복구한다.

- n 옵션일 때 복구할 파일의 새로운 파일명으로 복구한다.

**ls** : 리눅스 명령어 ls 와 같은 기능을 한다.

**vi(m)** : 리눅스 명령어 vi(m) 와 같은 기능을 한다.

**help** : 해당 프로그램에 내장명령어와 명령어의 옵션을 출력해서 보여준다.

**exit** : 해당 프로그램을 종료한다.

위에 내장 명령어들을 이용해 사용자가 백업을 원하는 파일이나 디렉토리를 옵션에 따라 추가 삭제하고 백업된 파일을 다시 복구하는 것을 관리하는 프로그램을 만들었다.

**2 . 기능**

char\* change\_path(char\* input\_path, char\* result\_path); // 입력받은 경로를 수정하는 함수

char\* get\_time(char\* tmp\_text); // 현재 시간 구하는 함수

int correct\_path(char\* path); // 입력받은 경로가 제대로 된 경로인지 확인하는 함수

int filecopy(const char\* input\_path, const char\* backup\_path); // 파일 복사하는 함수

int check\_hash\_code(char\* input\_path, char\* check\_path, char\* hash); // 해쉬함수로 파일 동일 여부 확인하는 함수

char\* return\_hash\_value\_md5(char\* path); // 파일 경로를 넣어줬을 경우 MD5함수를 이용해서 해쉬 값 나오는 함수

char\* return\_hash\_value\_sha(char\* path); // 파일 경로를 넣어줬을 경우 SHA1함수를 이용해서 해쉬 값 나오는 함수

char\* cut\_time\_path(char \*path, char\* temp); // backup path 에서 뒤에 14자리 자르는 함수

char\* cut\_path(char \*path, char\* temp, int backup\_path\_length); // backup path 랑 똑같게 자르는 함수

char\* return\_num(long num, char\* print\_num); // 숫자 입력시 숫자 3자리씩 끊어서 보여주는 함수

char\* cut\_backup\_path(char\* input\_path, char\* backup\_path); // 경로에서 "backup" 디렉토리 빼는 함수

char\* change\_input\_path(char\* input\_path, char\* save\_path, char\* new\_input); // 경로에서 "new input" 을 붙이는 함수

char\* change\_directory\_path(char\* input\_path, char\* save\_path, char\* new\_input); // 경로에서 "new input" 을 붙이는 함수

char\* make\_realpath(char\* path); // realpath 처럼 만들어주는 함수

void insert(char\* path, int file\_type); // 파일 관리 링크드 리스트로 만들기

void traverse\_directory(char\* dir\_path); // 링크드 리스트 재귀로 해당 디렉토리에 존재한는 파일 저장하는 함수

int count\_list(); // 링크드 리스트에 현재 존재하는 파일 개수 확인하는 함수

void print\_list(); // 링크드 리스트 값 앞에서 부터 확인하는 함수

void delete\_list(); // 링크드 리스트 삭제하는 함수

**3. 상세설계**

**[ ssu\_backup\_add.c ]**

**1. main**

- 첫번째 인자로 입력 되었는지 확인한다.

- 첫번째 인자를 절대경로로 바꿔서 input\_path 로 저장한다.

- input\_path 4,096byte 길이 확인한다.

**2. int correct\_path(char\* path);**

- input\_path 가 correct\_path() 함수로 실제로 존재하는 경로인지, 파일에 대한 접근 권한, 일반 파일 디렉토리인 지 확인한다.

- input\_path 경로에 홈 디렉토리가 포함되어 있는지 확인한다.

- input\_path 경로에 “backup” 경로가 포함되어 있는지 확인한다.

**3. void insert(char\* path, int file\_type);**

**void traverse\_directory(char\* dir\_path);**

- input\_path 에 대한 Insert(), traverse\_directory() 함수로파일을 링크드 리스트로 파일 관리한다.

- 해당 링크드 리스트를 재귀 순회 하면서 파일들을 저장한다.

**4. char\* change\_path(char\* input\_path, char\* result\_path);**

**char\* get\_time(char\* tmp\_text);**

- 파일의 유형을 확인한다

- 디렉토리일 경우 change\_path() 함수로 input\_path를 인자로 저장할 경로인 backup\_path로 바꾸고 해당 backup\_path에 디렉토리 생성한다.

- 파일일 경우 파일을 change\_path() 함수로 input\_path를 인자로 저장할 경로인 backup\_path로 바꾸고 해당 backup\_path 에 get\_time() 함수로 현재 시간 구한 후 backup\_path 에 경로를 추가해서 저장할 경로를 만든다.

**5. int check\_hash\_code(char\* input\_path, char\* check\_path, char\* hash);**

**char\* cut\_time\_path (char \*path, char\* temp\_path);**

- Backup directory 내에 cut\_time\_path() 함수로 backup directory에 현재 존재하는 파일들에서 시간부분을 제거하고 파일 경로를 만들어서 backup 할 파일과 경로가 같은지 확인한다.

**6. char\* return\_hash\_value\_md5(char\* path);**

**char\* return\_hash\_value\_sha(char\* path);**

**int filecopy(const char\* input\_path, const char\* backup\_path);**

- 해시값이 같은 파일이 있는지 여부 확인하고 존재하지 않을때 파일을 filecopy() 함수로backup\_path 에 backup 한다 .

**[ ssu\_backup\_remove .c ]**

**1. main**

**char\* make\_realpath(char\* path);**

- 첫번째 인자로 입력 되었는지 확인하기

- 첫번째 인자를 절대경로로 바꿔서 input\_path 로 저장한다.

- input\_path 4,096byte 길이 확인한다.

- make\_realpath() 함수로 backup\_path 를 만든다.

**2. void insert(char\* path, int file\_type);**

**void traverse\_directory(char\* dir\_path);**

- backup\_path 가 디렉토리인 경우

- backup\_path 에 대한 Insert(), traverse\_directory() 함수로파일을 링크드 리스트로 파일 관리한다.

- 해당 링크드 리스트를 재귀 순회 하면서 파일들을 저장한다.

- backup\_path 가 파일인 경우

- backup directory 에 대한 Insert(), traverse\_directory() 함수로파일을 링크드 리스트로 파일 관리한다.

- 해당 링크드 리스트를 재귀 순회 하면서 파일들을 저장한다.

**3. char\* change\_path(char\* input\_path, char\* result\_path);**

- input\_path 가 change\_path() 함수로 backup 되어있는 경로로 backup\_path 를 만든다

**4. int correct\_path(char\* path);**

- input\_path 가 correct\_path() 함수로 실제로 존재하는 경로인지, 파일에 대한 접근 권한, 일반 파일 디렉토리인 지 확인한다.

**5.**

- backup\_path 파일 유형 확인하기 파일 유형 확인해서 디렉토리일 경우

- backup\_path 파일이 디렉토리인 경우 옵션으로 -a 인지 확인한다.

**5 - 1. char\* cut\_path(char \*path, char\* temp, int backup\_path\_length); //**

- cut\_path 함수로 backup directory 내에 있는 파일들의 경로를 잘라서 backup\_path 와 같은지 확인한다.

- directory 내에 모든 파일을 삭제하고 해당 디렉토리도 삭제한다.

**6.**

- backup\_path 파일 유형 확인하기 파일 유형 확인해서 파일일 경우

- backup\_path 파일이 디렉토리인 경우 옵션으로 -a 인 경우

**6 - 1. char\* cut\_time\_path(char \*path, char\* temp);**

- cut\_time\_path 함수로 backup directory 내에 있는 파일들의 시간을 잘라서 backup\_path 와 같은지 확인한다.

- backup\_path 와 같은 파일들은 모두 삭제한다.

- 옵션이 없을 경우

**6 - 2. char\* return\_num(long num, char\* print\_num);**

- directory 내에 백업할때 경로가 같았던 파일이 존재할 경우에 해당 파일에 대한 어떤 파일을 선택해서 삭제해야하는지 파일 시간과 파일 크기를 return\_num() 함수로 출력한다.

**7.**

- 두번째 인자로 옵션으로 -c 일 경우 backup directory 내에 있는 모든 파일과 디렉토리를 모두 삭제한다.

- 삭제한 파일과 디렉토리를 확인해서 몇개씩 삭제했는지 출력한다.

**[ ssu\_backup\_recover.c ]**

**1. main**

**char\* make\_realpath(char\* path);**

- 첫번째 인자로 입력 되었는지 확인한다.

- 첫번째 인자를 절대경로로 바꿔서 input\_path 로 저장한다.

- input\_path 4,096byte 길이 확인한다.

- make\_realpath() 함수로 backup\_path 를 만든다.

**2. void insert(char\* path, int file\_type);**

**void traverse\_directory(char\* dir\_path);**

- backup\_path 가 디렉토리인 경우

- backup\_path 에 대한 Insert(), traverse\_directory() 함수로파일을 링크드 리스트로 파일 관리한다.

- 해당 링크드 리스트를 재귀 순회 하면서 파일들을 저장한다.

- backup\_path 가 파일인 경우

- backup directory 에 대한 Insert(), traverse\_directory() 함수로파일을 링크드 리스트로 파일 관리한다.

- 해당 링크드 리스트를 재귀 순회 하면서 파일들을 저장한다.

**3. char\* change\_path(char\* input\_path, char\* result\_path);**

- input\_path 가 change\_path() 함수로 backup 되어있는 경로로 backup\_path 를 만든다

**4. int correct\_path(char\* path);**

- input\_path 가 correct\_path() 함수로 실제로 존재하는 경로인지, 파일에 대한 접근 권한, 일반 파일 디렉토리인 지 확인한다.

- input\_path 4,096byte 길이 확인하기

- backup\_path 파일 유형 확인하기 파일 유형 확인하기

**5.**

- 파일 유형이 디렉토리인 경우 옵션 -d 인 경우

**5 - 1. char\* cut\_path(char \*path, char\* temp, int backup\_path\_length);**

- cut\_path 함수로 backup directory 내에 있는 파일들의 경로를 잘라서 backup\_path 와 같은지 확인한다.

**5 – 1 – 1. char\* cut\_backup\_path(char\* input\_path, char\* backup\_path);**

- 링크드 리스트에 저장된 파일들의 경로에서 cut\_backup\_path() 함수로 “backup” 경로를 제거한다 .

- “backup” 을 제거한 파일의 경로로 파일들의 유형을 확인한다.

- 파일 유형이 디렉토리인 경우 디렉토리 유무에 따라 디렉토리 생성한다.

- 파일 유형이 파일인 경우

**5 - 1 - 2.** **char\* cut\_time\_path(char \*path, char\* temp);**

- “backup” 제거한 경로에서 시간도 제거한다.

- backup directory 내에 recover 하려는 파일과 같은 파일 있는지 확인한다.

- 같은 파일이 없는 경우

**5 – 1 - 3. char\* return\_hash\_value\_md5(char\* path);**

**char\* return\_hash\_value\_sha(char\* path);**

**int filecopy(const char\* input\_path, const char\* backup\_path);**

- 해시값이 같은 파일이 있는지 여부 확인하고 존재하지 않을때 파일을 filecopy() 함수로 원래 파일 경로 에 recover 한다.

- 같은 파일이 존재할 경우

- 같은 파일에 대해 어떤 파일을 recover 할지 한번 물어보도록 다음에 같은 파일이 있을 경우 다음에 물어본다.

- 같은 파일들 중에서 recover 할 파일을 선택한다.

- **5 – 1 – 3** 을 재실행한다.

**5 – 1 - 4. char\* return\_num(long num, char\* print\_num);**

- recover 할 파일 이름과 return\_num() 함수로 파일 크기를 출력한다.

- 파일 유형이 디렉토리인 경우 옵션 -d, 옵션 -n, 새로운 바꿀 경로를 넣을 경우

- **5 - 2. char\* cut\_path(char \*path, char\* temp, int backup\_path\_length);**

- cut\_path 함수로 backup directory 내에 있는 파일들의 경로를 잘라서 backup\_path 와 같은지 확인한다.

**5 – 2 – 1. char\* cut\_backup\_path(char\* input\_path, char\* backup\_path);**

- 링크드 리스트에 저장된 파일들의 경로에서 cut\_backup\_path() 함수로 “backup” 경로를 제거한다 .

- “backup” 을 제거한 파일의 경로로 파일들의 유형을 확인한다.

- 파일 유형이 디렉토리인 경우

**5 – 2 – 2. char\* change\_directory\_path(char\* input\_path, char\* save\_path, char\* new\_input);**

- change\_directory\_path() 로 새로운 경로를 넣어준 경로로 디렉토리를 생성한다.

- 파일 유형이 파일인 경우

**5 - 2 - 3.** **char\* cut\_time\_path(char \*path, char\* temp);**

- “backup” 제거한 경로에서 시간도 제거한다.

- backup directory 내에 recover 하려는 파일과 같은 파일 있는지 확인한다.

- 같은 파일이 없는 경우

**5 – 2 - 4. char\* return\_hash\_value\_md5(char\* path);**

**char\* return\_hash\_value\_sha(char\* path);**

**int filecopy(const char\* input\_path, const char\* backup\_path);**

- 해시값이 같은 파일이 있는지 여부 확인하고 존재하지 않을때 파일을 filecopy() 함수로 새로운 경로를 넣어준 경로에 recover 한다.

- 같은 파일이 존재할 경우

- 같은 파일에 대해 어떤 파일을 recover 할지 한번 물어보도록 다음에 같은 파일이 있을 경우 다음에 물어본다.

- 같은 파일들 중에서 recover 할 파일을 선택한다.

- **5 – 2 – 3** 을 재실행한다.

**5 – 2 - 5. char\* return\_num(long num, char\* print\_num);**

- recover 할 파일 이름과 return\_num() 함수로 파일 크기를 출력한다

**6.**

- 파일 유형이 파일인 경우

**6 – 1**. **char\* cut\_time\_path(char \*path, char\* temp);**

- “backup” 제거한 경로에서 시간도 제거한다.

- backup directory 내에 recover 하려는 파일과 같은 파일 있는지 확인한다.

- 옵션 -n, 새로운 바꿀 경로를 넣을 경우

**6 – 1 – 1. char\* change\_input\_path(char\* input\_path, char\* save\_path, char\* new\_input);**

- change\_input\_path() 로 새로운 경로를 넣어준 경로로 파일 경로를 생성한다.

**6 - 1 - 2.** **char\* cut\_time\_path(char \*path, char\* temp);**

- “backup” 제거한 경로에서 시간도 제거한다.

- backup directory 내에 recover 하려는 파일과 같은 파일 있는지 확인한다.

- 같은 파일이 없는 경우

**6 – 1 - 3. char\* return\_hash\_value\_md5(char\* path);**

**char\* return\_hash\_value\_sha(char\* path);**

**int filecopy(const char\* input\_path, const char\* backup\_path);**

- 해시값이 같은 파일이 있는지 여부 확인하고 존재하지 않을때 파일을 filecopy() 함수로 원래 파일 경로 에 recover한다.

- 같은 파일이 존재할 경우

- 같은 파일들 중에서 recover 할 파일을 선택한다.

- **6 – 1 – 3** 을 재실행한다.

**6 – 1 - 4. char\* return\_num(long num, char\* print\_num);**

- recover 할 파일 이름과 return\_num() 함수로 파일 크기를 출력한다.

- 옵션이 없을 경우

**6 – 2**. **char\* cut\_time\_path(char \*path, char\* temp);**

- “backup” 제거한 경로에서 시간도 제거한다.

- backup directory 내에 recover 하려는 파일과 같은 파일 있는지 확인한다.

**6 – 2 – 1. char\* change\_input\_path(char\* input\_path, char\* save\_path, char\* new\_input);**

- change\_input\_path() 로 새로운 경로를 넣어준 경로로 파일 경로를 생성한다.

**6 - 2 - 2.** **char\* cut\_time\_path(char \*path, char\* temp);**

- “backup” 제거한 경로에서 시간도 제거한다.

- backup directory 내에 recover 하려는 파일과 같은 파일 있는지 확인한다.

- 같은 파일이 없는 경우

**6 – 2 - 3. char\* return\_hash\_value\_md5(char\* path);**

**char\* return\_hash\_value\_sha(char\* path);**

**int filecopy(const char\* input\_path, const char\* backup\_path);**

- 해시값이 같은 파일이 있는지 여부 확인하고 존재하지 않을때 파일을 filecopy() 함수로 원래 파일 경로 에 recover한다.

- 같은 파일이 존재할 경우

- 같은 파일들 중에서 recover 할 파일을 선택한다.

- **6 – 2 – 3** 을 재실행한다.

**6 – 2 - 4. char\* return\_num(long num, char\* print\_num);**

- recover 할 파일 이름과 return\_num() 함수로 파일 크기를 출력한다.

**[ ssu\_backup\_ls.c ]**

- 입력받은 인자에서 2번째 인자가 있는지 확인한다.

- 인자가 linux 명령어 “ls” 의 인자로 들어와서 실행한다.

**[ ssu\_backup\_vi.c ]**

- 입력받은 인자에서 2번째 인자가 있는지 확인한다.

- 인자가 linux 명령어 “vi” 의 인자로 들어와서 실행한다.

**[ ssu\_backup\_help.c ]**

- help 출력문 바로 출력한다.

**[ ssu\_backup.c ]**

- 프로그램 실행하면서 실행인자의 두번째 인자로 sha1 , md5 해시함수 종류를 넣어줬는지 확인한다.

- 실행 후 바로 현재 작업 디렉토리에 backup 디렉토리 생성한다.

- 학번 프롬프트 출력후 표준입력으로 받은 문자열 저장한다.

- 저장한 문자열에서 개행별로 token 을 잘라서 첫번째 토큰으로 어떤 함수를 실행할지 고른다.

- 개행 입력시 프롬프트 재출력한다.

- 모든 함수 내에서 첫번째 인자로 들어온 실행파일 위치를 저장한다.

- 모든 내장 명령어의 실행파일 경로에 앞에 붙혀서 해당 경로에 있는 실행파일을 실행한다.

- 첫번째 토큰 확인하고 해당 함수가 add, remove, recover 일 경우에 해시함수 종류를 해당 함수 실행할때 인자로 넣어준다.

- exit을 제외한 모든 내장 명령어는 fork() 함수를 이용해서 자식 프로세스 생성후 자식 프로세스에서 execv () 함수로 add() 함수가 있는 실행파일을 실행한다.

- 해당 자식 프로세스가 끝날때까지 부모 프로세스 정지한다.

- 지정된 내장명령어 외에 다른것을 입력할 경우 help 내장 명령어 실행한다.

- exit 내장 명령어는 프로세스 생성하지 않고 바로 exit 으로 프로그램 종료한다.

**상세 설계 – 모듈 구성도**

도표이(가) 표시된 사진

자동 생성된 설명

**상세 설계 - 순서도**

도표, 개략도이(가) 표시된 사진

자동 생성된 설명

**7**

**6**

**5**

**4**

**0**

**3**

**2**

**1**

도표이(가) 표시된 사진

자동 생성된 설명도표이(가) 표시된 사진

자동 생성된 설명도표이(가) 표시된 사진

자동 생성된 설명

**2**

**1**

**3**

**4**

도표이(가) 표시된 사진

자동 생성된 설명

**5**

도표이(가) 표시된 사진

자동 생성된 설명

**6**

도표이(가) 표시된 사진

자동 생성된 설명도표이(가) 표시된 사진

자동 생성된 설명

**7**

**0**

**4. 실행결과**

- ssu\_backup 명령어만 실행시 usage 출력

- ssu\_backup md5 로 실행

- 내장명령어 외에 명령어 예외처리

- help 내장명령어 실행

- 개행 입력시 바로 프롬프트 출력 예외 처리

- exit 내장명령어 실행

텍스트이(가) 표시된 사진

자동 생성된 설명

**[ ssu\_backup\_add.c ]**

- 첫 번째 인자 입력이 없거나 올바르지 않은 경로 (해당 파일이나 디렉토리)없을 경우 포함 경우 포함 입력 시Usage 출력 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

- 첫 번째 인자로 입력받은 경로() ( 절대 경로 가 사용자의 홈 디렉토리 $HOME 또는 를 ~) 벗어나거나 백업 디렉토리 혹은 백 업 디렉토리 내 파일 및 디렉토리를 포함한다면

“< 입력받은 경로 > can’t be backuped” 표준 출력 후 프롬프트 재출력





- 두 번째 인자로 올바르지 않은 옵션이 들어왔을 경우 출력 Usage 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- 첫 번째 인자가 디렉토리일 경우 ‘-d’ 옵션을 사용하지 않았다면 에러 처리 후 프롬프트 재출력



- 옵션으로 ‘-help’ 가 들어온 경우

텍스트이(가) 표시된 사진

자동 생성된 설명

-백업파일의 기존 경로가 같고, 해시값이 같은 파일이 이미 백업디렉토리에 존재한다면 백업을 하지 않고 백업파일의 “< 절대 경로 표 > is already backuped” 표준 출력 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- 파일에 대한 실행 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

- 디렉토리에 대한 실행 결과

- 디렉토리 내에 파일이 없을 경우 예외처리

텍스트이(가) 표시된 사진

자동 생성된 설명

- 디렉토리 내에 파일 중에 바뀐 파일만 backup 을 하고 바뀌지 않은 파일은 already backup 출력

텍스트이(가) 표시된 사진

자동 생성된 설명

**[ ssu\_backup\_remove.c ]**

- 위에 실행 후 디렉토리 구조

텍스트이(가) 표시된 사진

자동 생성된 설명

- 첫 번째 인자 입력이 없거나 올바르지 않은 경로 (해당 파일이나 디렉토리)없을 경우 포함 경우 포함 입력 시Usage 출력 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

- 첫 번째 인자로 입력받은 경로() ( 절대 경로 가 사용자의 홈 디렉토리 $HOME 또는 를 ~) 벗어나거나 백업 디렉토리 혹은 백 업 디렉토리 내 파일 및 디렉토리를 포함한다면

“< 입력받은 경로 > can’t be backuped” 표준 출력 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- 두 번째 인자로 올바르지 않은 옵션이 들어왔을 경우 출력 Usage 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- ‘-a’ 와 ‘-c’ 옵션이 동시에 들어왔을 경우 출력 Usage 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- 첫 번째 인자가 디렉토리일 경우 ‘-a’ 옵션을 사용하지 않았다면 에러 처리 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- ‘-c’ 옵션 사용 시 다른 인자를 입력하면 출력 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- 현재 backup directory 상태

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 에 파일 이름이 하나인 경우 remove



- 상대 경로 입력 시backup directory 에 파일 이름이 여러개인 경우 remove

텍스트이(가) 표시된 사진

자동 생성된 설명

- 절대 경로 입력 시backup directory 에 파일 이름이 여러개인 경우 remove

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 내부에 존재하는 디렉토리 remove

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 내부에 존재하는 이름이 같은 파일 한번에 remove

- 현재 상태

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

- 명령어 실행 후 상태

텍스트이(가) 표시된 사진

자동 생성된 설명

- 위의 명령어 모두 실행 후 backup directory



- remove -c 실행



- remove -c 실행 후 재실행



- 현재 backup directory 상태

텍스트이(가) 표시된 사진

자동 생성된 설명

**[ ssu\_backup\_recvover.c ]**

- 첫 번째 인자 입력이 없거나 올바르지 않은 경로 (해당 파일이나 디렉토리)없을 경우 포함 경우 포함 입력 시Usage 출력 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- 첫 번째 인자로 입력받은 경로 (절대 경로) 혹은 옵션 ‘-n’ 뒤에 <NEWFILE>인자가 사용자의 ( 홈 디렉토리 %HOME 또는 ~)를 벗어나거나 백업 디렉토리 혹은 백업 디렉토리 내 파일 및 디렉토리를 포함한다면 “< 입력 받은 경로 > can’t be backuped” 표준 출력 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

- 첫 번째 인자가 디렉토리일 경우 ‘-d’ 옵션을 사용하지 않았다면 에러 처리 후 프롬프트 재출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- 두 번째 인자로 올바르지 않은 옵션이 들어왔을 경우 Usage 출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 에 파일 이름이 하나인 경우 recover



- backup directory 에 파일 이름이 하나인 경우 recover 할 때 이미recover한 경우 “is already backuped recover” 출력



- 상대 경로 입력 시backup directory 에 파일 이름이 여러 개인 경우 recover

텍스트이(가) 표시된 사진

자동 생성된 설명

- 절대 경로 입력 시backup directory 에 파일 이름이 여러 개인 경우 recover

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 이미 내부에 존재하는 디렉토리 recover한 경우 “is already backuped recover” 출력



- backup directory 내부에 존재하는 디렉토리 recover 할 때 파일명이 같은 파일이 존재할 경우

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 에 파일 recover 할 때 옵션 ‘-n’ 을 사용한 경우 <NEWFILE> 인자 안 넣었을 경우 Usage 출력

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 에 파일 recover 할 때 옵션 ‘-n’ 을 사용하여 recover

텍스트이(가) 표시된 사진

자동 생성된 설명

- backup directory 에 디렉토리 recover 할 때 옵션 ‘-n’ 을 사용하여 recover

텍스트이(가) 표시된 사진

자동 생성된 설명

**[ ssu\_backup\_ls.c ]**

- ls실행 시

텍스트, 배터리, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**[ ssu\_backup\_vi.c ]**

- vi , vim실행 시



텍스트이(가) 표시된 사진

자동 생성된 설명

- exit실행 시

