1주차 예비보고서

전공 : 컴퓨터공학과 학년 : 2학년 학번 : 20211558 이름 : 윤준서

**1. 목적**

유닉스 시스템에 대하여 미리 접해본 후 실험에 임할 수 있도록 한다. 아울러 부록에 나 와 있는 명령어에 대하여 익숙해지도록 사용해본다.

**2. 예비학습**

2-1.

자신의 홈 디렉토리 주소를 불러오는 명령어인 echo $HOME 명령어를 통해 필자의 홈 디렉토리를 확인할 수 있다. 해당 명령어 사용 시 다음과 같은 디렉토리가 나오는데, 서강대학교 서버 내에 접속한 상태라 일반적인 홈 디렉토리 형태인 /home/ 이 아닌 /under/ 가 출력되었다.



2-2.

데이터 파일을 형성, 편집하기 위해 vi data 를 입력하여 파일을 형성하고, 사전에 다운받은 pc의 data 파일에 있는 목록을 옮겨 적는다. 그럼 다음과 같은 내용의 파일이 만들어진다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2-3.

vi data 를 통해 파일을 생성한 후, 명령모드에서 내용 입력을 위해 명령어 i 를 입력해 삽입모드로 전환한다. 이후 esc 를 입력, 마지막 행 모드로 전환 후 저장 후 종료를 위한 명령어 :wq 를 입력해 저장 후 디렉토리로 이동한다.

2-4.

파일을 복사하기 위해 cp 명령어를 이용한다.

data 를 통해 $home/.data 를 복사 생성하기 위해 cp data $home/.data 를 입력한다.

그럼 $home/.data 가 생성되는데, 알맞게 복사되었는지 확인하기 위해 텍스트 파일 명령어 nano 를 사용한다. nano $home/.data 를 입력하면 해당 파일의 내용을 확인할 수 있다.

텍스트, 스크린샷, 폰트, 디스플레이이(가) 표시된 사진

자동 생성된 설명

2-5.

그룹과 다른 사용자가 $HOME/.data 파일에 아무 권한을 갖지 않도록 하기 위해선 chmod 명령어를 사용한다. 이미 갖고 있는 권한을 삭제하는 것이 더 빠르므로, chmod [대상]-[권한 종류] [파일] 의 형태를 이용하여 권한을 삭제할 수 있다. 우린 삭제할 [대상]이 그룹(g)과 다른 사용자(o)이고, 모든 종류의 권한을(읽기, 쓰기, 실행) $HOME/.data 파일에 적용하려 하므로

chmod go-rwx $HOME/.data 를 입력한다. 이때 g(그룹) o(다른 사용자)이고, r(읽기) w(쓰기) x(실행)을 의미한다. 다음과 같이 사용자(u)만 권한이 생긴 것을 확인할 수 있다.





2-6.

권한을 설정하는 방법은 앞서 말한 권한 삭제 방법의 반대인 chmod [대상]+[권한 종류] [파일] 를 이용한다. '+'와 '-'의 차이를 주의해야 한다. '+'는 권한을 부여하는 것이고, '-'는 권한을 박탈하는 것이다. '='의 경우 [대상]에게 권한을 부여하고 나머지 [대상]과 [권한 종류]를 모두 박탈한다. 이때 설정하는 권한의 종류는 읽기(r), 쓰기(w), 실행(x) 의 세 종류다. 그리고 대상이 파일과 그 하위 디렉토리 두 가지로 나뉜다.

• 읽기(r)

파일을 확인, 열 수 있다.

파일의 목록을 보여주는 명령어인 ls 를 통해 하위 디렉토리 내의 파일의 존재를 확인할 수 있다.

• 쓰기(w)

파일에 작업(생성, 수정)을 수행할 수 있다. 단 파일의 이름 변경과 삭제는 불가능하다.

하위 디렉토리 내의 파일들은 위와 다르게 이름 변경과 삭제가 가능하다. 대신 파일 수정이 불가능하다.

• 실행(x)

파일을 실행할 수 있다.

하위 디렉토리로 이동할 수 있다.

**3. 보충학습**

정규 표현식이라고도 불리는 Regular Expression 은 데이터 검색 등에 사용되는 특수한 문자이다.

여러 데이터들이 나열되어 있는 파일에서 원하는 종류의 데이터를 찾거나 뽑을 때 사용된다.

데이터의 양이 많을 수록 유용한데, 예를 들어 데이터 파일 내에서 알파벳 d 가 2개 포함된 데이터만 찾고 싶을 때 Regular Expression 을 통해 쉽게 찾을 수 있다.

이 Regular Expression 은 다시 기본, 간격, 확장의 세 가지로 나뉜다.

• Basic Regular Expression 기본 정규 표현식

" " 를 통해 사용한다. 예를 들어 표현식 ^ 의 경우 문자열의 첫 글자를 검색한다. "^d" 의 경우 dragon, domain 과 같은 d 로 시작하는 데이터들을 검색하는 것이다. 문자열의 끝 글자를 검색하는 표현식 $ 를 사용하는 경우엔 "d$" 를 통해 solid, greed 와 같은 d 로 끝나는 데이터들을 검색할 수 있다.

다음은 Basic Regular Expression 의 종류들이다.

^ : 문자열의 첫 글자를 검색한다.

$ : 문자열의 끝 글자를 검색한다.

. : 점의 개수 만큼 임의의 문자로(공백 포함) 대체하여 검색한다.

\* : \* 앞의 문자를 검색한다.

₩ : 특수 문자를 검색한다.

? : 정확한 한 개의 문자를 검색한다.

• Interval Regular Expression 간격 정규 표현식

문자열 내에서 특정 문자가 몇 개가 포함되어 있는지를 이용해서 검색한다. 특정 패턴을 검색할 때 유용하다. { } 를 통해 사용하는데, 예를 들어 acceleration 과 같은 c 가 3개 포함된 데이터들을 검색할 때 "c" {3} 를 통해 검색이 가능하다.

다음은 Interval Regular Expression 의 종류들이다.

{n} : 앞의 문자가 n 개 있는 데이터를 검색한다.

{n, m} : 앞의 문자가 n 개 있는 데이터를 검색하는데, m 개 이하로 검색한다.

{n, } : 앞의 문자가 n 개 이상으로 있는 데이터를 검색한다.

• Extended Regular Expression 확장 정규 표현식

문자들의 순서, 조합을 이용하여 검색한다. 예를 들어 b 앞에 a 가 있는 데이터들을 검색할 때

"a\+b" 를 이용하여 absorption, abandon 과 같은 데이터를 검색할 수 있다.

다음은 Extended Regular Expression 의 종류들이다.

\+ : \+ 앞의 문자가 한 번 이상 나오는 데이터를 검색한다.

\? : \? 앞의 문자가 한 번 이하 나오는 데이터를 검색한다.