유닉스 프로그래밍 프로젝트 1 과제

컴퓨터과학과 201810954 안지민

주제 설명

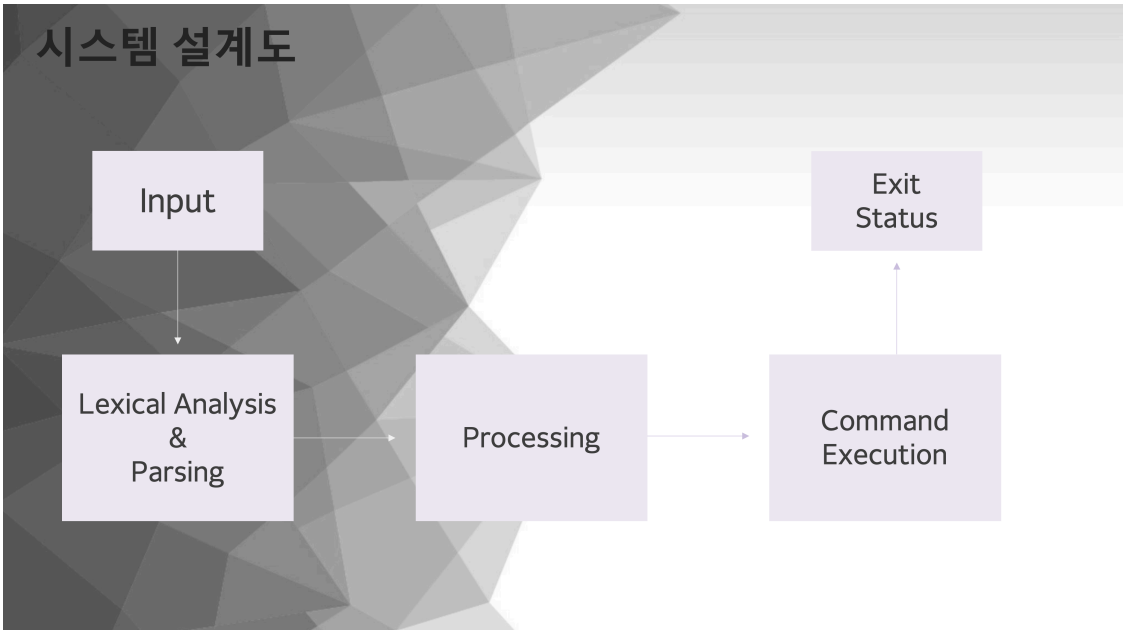
주제

리눅스 환경에서 나만의 셸(SMUshell) 만들기

설명

shell 이란 표준 입력으로부터 커맨드 라인을 읽고, 입력된 라인을 해석하여 이에 해당하는 명령을 실행시킨다. shell 의 가장 간단한 동작방식은 명령을 표준 입력으로부터 읽어들인 후, 명령을 실행시킬 자식 프로세스를 fork 하는 것이다. 그 후 부모 프로세스는 다른 명령을 읽어들이기 전에 자식 프로세스가 종료할 때까지 기다린다. 또한 shell 은 파이프 라인과 재지향 외에 시그널을 처리하며 본 과제는 이러한 shell 을 부분적이지만 전체를 아우를 수 있는 스뮤셸(SMUshell)을 작성하는 것으로 초점을 두었다.

시스템 설계도



사용된 기술

시그널 처리

일반적으로 수행되는 프로세스의 경우 Ctrl+c(SIGINT) 시그널을 통하여 수행 중인 프로세스를 종료시킬 수 있지만, SMUshell 의 경우엔 이러한 시그널을 무시하도록 처리하였다. SMUshell 의 종료는 exit 이라는 셸 내부 명령어를 통해서만 가능하도록 처리하였다

파이프 라인

(Implementation of multipe pipes in C 을 참고하여 smu_shell.c 의 recurPipe()함수를 작성하였습니다)

original parent process 는 각 명령어를 위해서 child 를 생성하므로 ordeal 내내 살아남을 수 있으며 자식은 이전 명령에서 입력을 받아야 하는지, 출력을 다음 명령으로 보내야 하는지 여부를 확인한다. 이후 pipe file descriptor 의 모든 복사본을 모두 닫은 후 실행한다. 부모는 명령마다 자식을 생성할 때까지 fork 외에는 아무것도 하지 않으며 이후, 부모는 descriptor 의 모든 복사본을 닫고 wait 를 할 수 있다.

필요한 파이프를 모두 만든 다음 루프에서 관리를 위해 배열 pipefds[]를 사용하였다.

재지향

smu_shell.c 의 myexecu()함수를 사용해서 redirection 명령어가 들어올 경우 처리하였다. 명령 인자에서 '>', '<' 여부를 각각 확인하고 전자가 들어왔을 경우 파일 쓰기를 실행, 후자가 들어왔을 경우 파일 읽기를 실행하였다. 또한 해당 함수 내에서 그 외에 명령어가 들어왔을 경우, 시스템에서 제공하는 함수들을 사용하여 처리해주었다.

사용 메뉴얼

SMUshell 의 파일 구성

1. smu_shell.c : 메인 함수 및 shell 의 기능을 구현한 함수들(파이프라인, 재지향)
2. support.c : 연결 리스트 구조체 관련 기타 함수들
3. support.h : 헤더 파일 및 연결리스트 구조체, 함수 프로토 타입 정의
4. Makefile
    - $make : support.c, smu_shell.c 코드 컴파일
    - $clean : ₩rm *.o smu_shell 수행
    - $./smu_shell : make 로 컴파일 후 코드 실행 가능

SMUshell 의 실행 과정

프롬프트가 뜨고, 첫 실행 시 웰컴 메세지가 뜨게 된다(버전 정보, Author, University ID, Date 포함)

그 후 입력 받은 명령들은 fork() 함수를 통하여 자식 프로세스에서 수행된다. 부모 프로세스의 경우는 자식 프로세스의 동작이 완료될 때까지 wait 했다가 수행이 종료되면 다시 프롬프트를 띄우는 역할을 하게 된다. fork()함수를 통하여 생성된 자식 프로세스는 명령어들을 분리한 argument vector table 에 의하여 exec 계열의 함수를 통하여 실행이 된다. 통상적으로 쓰이는 함수는 execv(), execvp()인데, SMUshell 에서는 실행 파일의 절대 경로를 포함하는 execv()함수만 사용하였다

SMUshell 의 명령어

SMUshell 은 기본 리눅스 셸에서 사용하는 일반적인 명령어들을 입력하여 실행시킬 수 있으며, 몇 가지 셸 내부 명령어는 직접 작성한 함수 형태로 실행하게 된다.

재정의한 셸 내부 명령어의 종류는 다음과 같다

path, cd, exit

단 SMUshell 내에서 몇 가지 구현되지 않은 명령어가 있는데 대표적으로 Expansion(Brace, Tilde)이다. shell 의 핵심적인 기능 구현을 우선으로 하다보니 시간관계 상 해당 부분은 구현하지 못했다.

사용 예

$make 명령어를 통한 컴파일

```
(base) mac@MACui-iMac UNIX_Myshell % make
gcc -c support.c
gcc -c smu_shell.c
gcc -o smu_shell support.o smu_shell.o
```

$./smu_shell 명령을 통해 SMU shell 실행

- 웰컴 메세지와 함께 부가적인 정보가 뜸
- 그림은 SMU 의 상징동물인 사슴을 표현한 것
- 명령어를 입력하는 SMUshell$이 뜸

```
(base) mac@MACui-iMac UNIX_Myshell % ./smu_shell
< Hi, welcome to SMU Shell! >
                \
                 \   \_\_    _/_/         Version 1.1
                  \      \__/               Author. jimin An
                   \     (oo)\_______         UNI ID. 201810954
                    \    (__)\       )\/\     Date. 2020/11/07
                              ||----- |
                              ||     ||

SMUshell$
```

$cd, $pwd 명령어

```
SMUshell$cd ..
SMUshell$pwd
/Users/mac
SMUshell$cd UNIX_Myshell
SMUshell$pwd
/Users/mac/UNIX_Myshell
```

pipe line

```
SMUshell$ls
Makefile        myshell         smu_shell.c     support.c       support.o
README.md       smu_shell       smu_shell.o     support.h
SMUshell$ls | grep c
smu_shell.c
support.c
```

Redirection

'>' : 프로그램의 결과를 파일로 저장하기

```
SMUshell$ls
Makefile        myshell         smu_shell.c     support.c       support.o
README.md       smu_shell       smu_shell.o     support.h
SMUshell$ls > text.txt
SMUshell$cat text.txt
Makefile
README.md
myshell
smu_shell
smu_shell.c
smu_shell.o
support.c
support.h
support.o
text.txt
```

$exit 을 통한 SMUshell 종료

- 시그널 처리를 통해 Ctrl+c 를 통한 종료는 불가, 오직 exit 내부 명령어를 통해서 종료 가능

```
SMUshell$^C
Command  not found.
SMUshell$ exit
Closing...
(base) mac@MACui-iMac UNIX_Myshell %
```