

Project #1: Unix Utilities

Objectives

- To be familiar with the C programming language and Linux environment.

1. My-cat

- 1) 아래와 같이 my-cat.c 를 작성했다.

```
#include <stdio.h>
#include <stdlib.h>
#define LINE_SIZE 100

int main(int argc, char *argv[]){

    int i;
    FILE *fp[argc-1]; //file pointer
    char str_buf[LINE_SIZE];

    if (argc == 1){ //명령어만 있고 file 명을 입력하지 않았을 때
        printf("There is no file\n");
        exit(1);
    }
    for(i =1; i < argc ; i++) //명령어 뒤에 있는 file 을 읽어서 print 하는 과정
    {
        fp[i] = fopen(argv[i],"r"); //file open

        if (fp[i] == NULL){ //파일을 못 열었을 때 경고 메세지
            printf("my-cat: cannot open file\n");
            exit(1);
        }
        else{ //열린 파일의 텍스트를 읽어서 print 함
            while(1){ //한 줄 씩 str_buf 에 저장하고 print 함
                if(fgets(str_buf, LINE_SIZE,fp[i]) != NULL){
                    printf("%s", str_buf);
                }
                else
                    break;
            }
            fclose(fp[i]); //close file
        }
    }
    exit(0);
}
```

: argc로 argument의 수를 세고 FILE 포인터를 이용해 첫 번째 argument(실행 파일)를 제외하고 파일로 받아드린다. argc = 1 일 때, file 을 입력하지 않았으므로 "There is no file"이라는 경고 메시지를 띄워준다. 이 if 문을 지나쳤을 때는 오류가 없기때문에 file 을 읽어서 print 하는 과정을 file 이 모두 열릴 때까지 반복한다. 자세한 내용은 주석으로 설명했다.

2) 컴파일

: 에러 없이 컴파일이 됐고 my-cat 이라는 실행 파일이 생성된 것을 볼 수 있다(붉은색).

```
Sungminui-MacBook-Pro:proj1 sungminryu$ gcc -o my-cat my-cat.c -Wall -Werror
Sungminui-MacBook-Pro:proj1 sungminryu$
```

```
Sungminui-MacBook-Pro:proj1 sungminryu$ ls
bar.txt  foo.txt  my-cat  my-cat.c  my-grep  my-grep.c  test.txt
```

3) my-cat 실행(with bar.txt)

: bar.txt 가 그대로 출력되었다.

```
Sungminui-MacBook-Pro:proj1 sungminryu$ ./my-cat bar.txt
ToDo :
1. Make my-cat
2. Make my-grep
3. Make my-zip and my-unzip
Go to Sleep!!
Sungminui-MacBook-Pro:proj1 sungminryu$
```

4) "./my-cat"를 cat 이라는 명령어로 Built-in 시킨 후 두 가지 텍스트 파일 오픈

: 다수의 파일을 열었을 때도 잘 동작되었다.

```
Sungminui-MacBook-Pro:proj1 sungminryu$ alias cat="./my-cat"
Sungminui-MacBook-Pro:proj1 sungminryu$ alias
alias cat='./my-cat'
Sungminui-MacBook-Pro:proj1 sungminryu$ cat bar.txt foo.txt
ToDo :
1. Make my-cat
2. Make my-grep
3. Make my-zip and my-unzip
Go to Sleep!!
this line has foo in it
so does this foolish line; do you see where?
hello
even this line, which has barfood in it, will be printed.
football
Sungminui-MacBook-Pro:proj1 sungminryu$
```

1. My-grep

1) 아래와 같이 my-grep.c 를 작성했다.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //string 비교를 위해 추가
#define LINE_SIZE 100

int main(int argc, char *argv[]){

    FILE *fp; //file pointer(한 개의 파일만 읽음)
    char str_buf[LINE_SIZE];

    if(argc == 2){ //파일 없이 찾으려는 단어만 입력했을 때
        fp = stdin; //standard input 을 file 로 받음
        goto jump; //jump 로 이동
    }

    if (argc != 3){ //grep word file 형식이 아닐 경우 오류 메세지
        printf("my-grep: searchterm [file ...]\n");
        exit(1);
    }

    fp = fopen(argv[2],"r"); //입력한 파일을 open

    jump: //standard input 일 때 argv[2]가 없으므로 skip

    if (fp == NULL){ //파일을 못 열었을 때 경고 메세지
        printf("my-grep: cannot open file\n");
        exit(1);
    }

    else{ //열린 파일의 텍스트를 읽어서 print 함
        while(1){ //한 줄 씩 str_buf 에 저장
            if(fgets(str_buf, LINE_SIZE,fp) != NULL){
                //찾으려는 단어가 포함되어 있는 줄만 print
                if(strstr(str_buf, argv[1]) != NULL){
                    //strstr: 두 개의 string 을 비교해서 일치하면 1 아니면 0 을 반환
                    printf("%s", str_buf);
                }
            }
            else
                break;
        }
        fclose(fp);
        exit(0);
    }
}
```

: my-cat 과 같은 방법으로 argc 로 argument 의 수를 센다. my-cat 과 다른 점은 FILE 포인터를 이용해 세 번째 argument(실행 파일)만 파일로 받아드린다. 따라서 FILE 포인터를 하나만 잡아주었다(my-cat 에서는 array 의 포인터). 즉, 두 번째 argument 는 찾으려는 단어이고 세 번째가 파일이 된다.

경우의 수는 다음과 같이 나뉜다.

- Argument 가 2 인 경우(\$grep foo)
파일이 없으므로 standard input 을 파일로 생각한다(fp=stdin). File open 을 넘어가야하므로 goto 문을 이용해 jump 로 이동시킨다.
- Argument 가 3 인 경우(\$grep foo foo.txt)
파일을 open 한다(fp = fopen(argv[2], "r")).
- 그 외
오류 메시지("my-grep: searchterm [file ...]\n")

위의 경우의 수 판단하는 if 문을 지나오면 두 번째 argument(찾으려는 단어)와 일치하는 file 안의 행을 찾아야한다. 따라서 string.h 라이브러리에 있는 strstr() 함수를 이용해 찾으려는 단어가 있을 때만 출력해준다. 자세한 내용은 주석으로 설명했다.

2) 컴파일

```
Sungminui-MacBook-Pro:proj1 sungminryu$ gcc -o my-grep my-grep.c -Wall -Werror
Sungminui-MacBook-Pro:proj1 sungminryu$
```

```
Sungminui-MacBook-Pro:proj1 sungminryu$ ls
bar.txt  foo.txt  my-cat  my-cat.c  my-grep  my-grep.c  test.txt
```

3) my-grep 실행(find "foo" in foo.txt)

: "foo"가 있는 1, 2, 4, 5 줄만 출력

```
Sungminui-MacBook-Pro:proj1 sungminryu$ cat foo.txt
1 this line has foo in it
2 so does this foolish line; do you see where?
3 hello
4 even this line, which has barfood in it, will be printed.
5 football
Sungminui-MacBook-Pro:proj1 sungminryu$ ./my-grep foo foo.txt
1 this line has foo in it
2 so does this foolish line; do you see where?
4 even this line, which has barfood in it, will be printed.
5 football
Sungminui-MacBook-Pro:proj1 sungminryu$
```

- 4) `"/.my-grep"`을 `grep` 이라는 명령어로 Built-in 시킨 후 standard input 확인
: 입력 받은 string 에 `"foo"`가 있을 경우만 출력

```
Sungminui-MacBook-Pro:proj1 sungminryu$ alias grep="/.my-grep"
Sungminui-MacBook-Pro:proj1 sungminryu$ alias
alias cat="/.my-cat"
alias grep="/.my-grep"
Sungminui-MacBook-Pro:proj1 sungminryu$ grep foo
I'm sungmin
Nice to meet you!
v I like football.
v I like football.
v foo
v foo
v I love eating food.
v I love eating food.
see you~
█
```