**< 파일에서 검색하기 - 코드 설명>**

- 헤더파일

표준 입출력, 입출력함수를 사용하기 위한 헤더파일 **<stdio.h>**

유닉스에서 사용하는 c 컴파일러 헤더파일 **<unistd.h>**

함수 조회와 설정을 할 수 있는 함수를 포함한 헤더파일 **<fcntl.h>**

문자열 변환 등 이용되지만 exit()함수를 사용하기 위해 포함한 헤더파일 **<stdlib.h>**

c형식 문자열을 다룰 수 있는 함수를 포함한 헤더파일로 strlen()함수를 위해 포함 **<string.h>**

**int main(void)**

**int rfd, n;**

읽기전용 파일 디스크립터 변수를 표현하기 위한 변수 rfd

글자 수를 의미하는 변수 n

**char buf[256];** 버퍼의 크기를 256으로 지정

**off\_t start, cur;** off\_t를 이용하여 파일 오프셋을 이동시킬 기준 값을 지정

**int count\_n =0;** 찾은 검색어의 횟수를 나타내기 위한 count\_n 변수로, 0으로 초기화 시켜준다.

**rfd = open("data2.txt", O\_RDONLY);**

open( ) 함수를 이용하여 txt파일을 열어 달라고 요청

O\_RDONLY 파일을 읽기 전용으로 연다.

**if (rfd == -1) {**

**perror("Open data2.txt");**

**exit(1);**

**}**

현재 열려 있는 파일을 구분하는 정수 값인 파일 기술자는 0 ‘표준 입력’ 부터 시작한다.

-1일 경우에는 파일이 제대로 읽혀 지지 않는다는 의미와 같으므로 -1일 경우 에러문 출력

perror( ) : 존재하지 않는 파일을 읽기 모드로 열었을 경우 에러 메시지를 출력하는 함수이다. exit( ) : 현재의 c프로그램 자체를 완전히 종료 시키는 기능을 가지는 함수

- ( )안에 1을 넣는 이유는 에러가 발생하여 종료할 때 대체로 1를 반환한다.

**start = lseek(rfd, 0, SEEK\_SET);**

lseek( ) : 함수의 seek pointer 커서를 조정하는 함수로,

rfd = 조정할 파일의 파일 디스크립터

0 = off\_t offset으로 기준점으로부터 이동할 거리

SEEK\_SET = 파일의 맨 앞으로 기준점을 지정해준다

즉 rfd 파일의 맨 앞으로 커서를 조정한다.

**n = read(rfd, buf, sizeof(buf));**

read( ) : open()함수로 열기로 한 파일의 내용을 읽는 함수

rfd = 읽을 파일의 파일 디스크립터

buf = 파일을 읽어 들일 버퍼

sizeof(buf) = 버퍼의 크기 sizeof( ) 함수는 괄호 안에 자료형의 크기를 출력하는 함수

즉, rfd 파일을 버퍼의 크기만큼 버퍼에 읽어 들인다.

**for (n = 0; n < strlen(buf); n++) {**

**if (buf[n] == 'a') {**

**count\_n++;**

**}**

**}**

반복문 for ( 범위 ) { 실행문 }

: n이 문자열의 길이만큼 반복문을 도는 동안 만약 ‘a’라는 검색어가 나올 경우

strlen( ) : c언어 스타일의 문자열을 받아 그것의 길이를 반환하는 함수

증감연산자 ++ : 값을 1 증가시킨 후 연산을 진행시키는 연산자

즉, n이 문자열의 길이만큼 반복문을 도는 동안 만약 ‘a’라는 검색어 나올 경우 count\_n 증감.

**buf[n] = '\0';**

문자열 제일 끝을 표현해주기 위해 ‘\0’을 넣어서 표현한다.

**printf("find word = % d\n”, count\_n);**

반복문을 통해 찾은 검색어의 수 출력.

**close(rfd);**

열 수 있는 파일의 개수는 제한 되어있으므로 파일을 닫아준다.

**return 0;**

main함수를 끝낸 후 벗어나기 위한 return