컴퓨터학부 20162518 최승서

1. **개요**
   1. **draw.c**

draw.c는 사용자로부터 이등변의 길이 이등변을 표현할 기호, 이등변을 저장할 파일 이름 이렇게 3가지를 입력을 받는다. 그리고 사용자가 입력한 기호로 입력한 크기의 이등변을 그려서 입력한 파일이름으로 저장하는 프로그램이다.

사용자 입력이 들어오면 우선 해당하는 이름의 파일을 쓰기 모드로 open 을 한다. 이때 만약 파일이 이미 존재하면 모든 내용을 초기화한다.

다음으로 가장 바깥쪽 for문을 사용자가 입력한 이등변의 길이만큼 수행하게 하여 이등변의 높이를 정한다. 이때 음수가 들어올 수 있음으로 abs()함수를 이용하여 양의 정수로 반복문을 수행하게 된다. 해당 반복문 내부에는 사용자가 입력한 이등변의 길이가 음수인지 양수인지 확인하여 이등변 삼각형을 거꾸로 출력할지 분기한다. 이때 말하는 출력은 파일에 write되는 것을 의미한다.

일련의 과정이 완료되면 close()를 통해 fd를 닫아준다.

* 1. **upperLower.c**

upperLower.c는 사용자가 존재하는 파일 이름을 입력하면 해당 텍스트 파일을 읽어서 소문자는 대문자로 대문자는 소문자로 변환하는 프로그램이다. 변환된 문자들은 입력한 파일이름도 대,소문자를 변환한 텍스트 파일을 생성해 저장한다.

먼저 입력이 들어오면 사용자가 입력한 파일을 읽기 모드로 open()한다. 다음으로 입력한 이름을 대소문자 변환을 시켜 변환한 이름으로 쓰기 모드로 파일을 open()한다. 이때 O\_CREAT, O\_TRUC옵션을 넣어서 파일을 생성하고 이미 존재한다면 초기화하여 사용한다.

다음으로 기존 파일의 텍스트를 write()를 이용하여 100바이트 단위로 읽어서 대소문자 변환 후 쓰기 모드로 open()한 파일에 write()를 한다.

일력의 과정이 완료되면 close()를 통해 open()한 2개의 파일을 닫아준다.

* 1. **randSort.c**

randSort.c는 사용자가 몇 개의 랜덤한 숫자를 생성할지 입력하면 입력한 숫자만큼의 랜덤으로 생성된 숫자를 number.txt에 write()하고 다시 number.txt를 읽어서 랜덤으로 생성된 숫자들을 정렬한 후 sort.txt에 write()하는 프로그램이다.

사용자 입력이 들어오면 해당 입력이 10 이상인지 확인한 후 number.txt파일을 읽기쓰기 모드로 open()을 한다.

다음으로 사용자가 입력한 숫자만큼의 랜덤한 숫자를 rand()를 이용하여 생성한다. 이때 매번 다른 랜덤한 숫자를 입력 받기 위해 srand를 사용하였다. 그리고 rand()를 통해 생성된 숫자에 %100을 하여 100자리 이하의 숫자만 생성하도록 제한하였다. 이렇게 생성된 숫자를 sprintf()함수를 이용하여 char형 buffer에 저장하여 숫자끼리 구분을 위해 buffer 배열 끝에 “ “를 추가하였다. 매번 이런 방식을 통해 생성될 때 마다 number.txt.에 write()를 해준다.

이제 number.txt에 저장된 숫자들을 한번에 읽어오기 위해 lseek()를 사용하여 해당 파일의 바이트 사이즈를 알아낸다. 그런 후 파일의 바이트 사이즈만큼의 char형 배열 to\_read를 사이즈만큼 동적 할당한다. 그리고 정렬할 숫자들을 저장할 int 형 배열 array도 사용자가 입력한 수만큼 동적 할당한다. 다음으로 파일사이즈만큼의 buffer에 파일 내용을 한번에 읽어온다. 그리고 읽어온 buffer를 “ “기준으로 토큰화하여 생성된 랜덤 숫자들을 int형 array에 저장한다. 이 array에 저장된 숫자들을 내림차순으로 정렬한다.

이렇게 정렬된 숫자들을 sort.txt.를 쓰기 모드로 opne()하여 정렬된 숫자들을 write()해준다.

일련의 과정이 완료되면 close()를 통해 number.txt, sort.txt를 닫아준다.

1. **소스코드와 주석**

**2-1. draw.c**

#include<stdio.h>

#include<stdlib.h>

#include<fcntl.h>

#include<unistd.h>

int main(){

int fd;

int side;

char ch;

char \*name;

printf("input :");

scanf("%d %c %s", &side, &ch, name);

if((fd = open(name, O\_WRONLY | O\_CREAT | O\_TRUNC, 0666)) <0 ){

fprintf(stderr, "open error for %s\n", name);

}

for(int i= 0; i<abs(side); i++){

if(side > 0){

for(int j = 0; j<i+1; j++)

write(fd, (char \*)&ch, sizeof(ch));

}else{

for(int j=abs(side)-i; j>0; j--)

write(fd, (char \*)&ch, sizeof(ch));

}

write(fd, "\n", sizeof(char));

}

close(fd);

return 0;

}

**2-2. upperLower.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<fcntl.h>

#include<unistd.h>

#define MAX\_NAME\_SIZE 20

#define MAX\_BUF\_SIZE 100

void abcChange(char \*str){

for(int i=0; i<strlen(str); i++){

if(str[i] >= 'a' && str[i] <= 'z')

str[i]-=32;

else if(str[i] >= 'A' && str[i] <= 'Z')

str[i]+=32;

}

}

int main(){

int fd1;

int fd2;

int read\_size;

char buffer[MAX\_BUF\_SIZE] = {0, };

char name[MAX\_NAME\_SIZE];

printf("input :");

scanf("%s", name);

if((fd1 = open(name, O\_RDONLY)) <0){

fprintf(stderr, "open error for %s\n", name);

exit(1);

}

abcChange(name);

if((fd2 = open(name, O\_WRONLY | O\_CREAT | O\_TRUNC, 0666)) <0){

fprintf(stderr, "open error for %s\n", name);

exit(1);

}

while((read\_size = read(fd1, buffer, MAX\_BUF\_SIZE)) > 0 ){

abcChange(buffer);

write(fd2, buffer, read\_size);

}

close(fd1);

close(fd2);

return 0;

}

**2-3. randSort.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<fcntl.h>

#include<unistd.h>

#include<time.h>

void write\_rand(int num, int fd){

int cnt, random\_number;

char buffer[10];

for(int i=0; i<num; i++){

random\_number = rand()%100;

cnt = sprintf(buffer, "%d", random\_number);

buffer[cnt] = ' ';

write(fd, buffer, cnt+1);

memset(buffer, '\0', 10);

}

}

void write\_sort(int fd, int\* array, int num){

int cnt;

char buffer[10];

for(int i=0; i<num; i++){

cnt = sprintf(buffer, "%d", array[i]);

buffer[cnt] = ' ';

write(fd, buffer, cnt+1);

memset(buffer, '\0', 10);

}

}

int get\_file\_size(int fd){

int size = lseek(fd, 0, SEEK\_END);

lseek(fd, 0, SEEK\_SET);

return size;

}

void save\_array(char \*str, int\* array){

char \*cut = strtok(str, " ");

int i=0;

while(cut != NULL){

array[i] = atoi(cut);

cut = strtok(NULL, " ");

i++;

}

}

void sort(int\* array, int cnt){

int tmp;

for(int i=0; i<cnt; i++){

for(int j=0; j<cnt-1; j++){

if(array[j]<array[j+1]){

tmp = array[j];

array[j] = array[j+1];

array[j+1] = tmp;

}

}

}

}

int main(){

//rand()함수에서 매번 다른 숫자를 출력할 수 있게 srand 사용

srand((unsigned int) time(NULL));

int fd1;

int fd2;

int number;

printf("INPUT : ");

scanf("%d", &number);

if(number >= 10){

if((fd1 = open("number.txt",O\_RDWR | O\_CREAT | O\_TRUNC, 0666)) <0){

fprintf(stderr, "open error for %s\n", "number.txt");

exit(1);

}

write\_rand(number, fd1);

int size = get\_file\_size(fd1);

char \*to\_read = (char \*)malloc(sizeof(char)\*size);

int \*array = (int \*)malloc(sizeof(int)\*number);

read(fd1, to\_read, size);

save\_array(to\_read, array);

sort(array, number);

if((fd2 = open("sort.txt",O\_WRONLY | O\_CREAT | O\_TRUNC, 0666)) <0){

fprintf(stderr, "open error for %s\n", "sort.txt");

exit(1);

}

write\_sort(fd2, array, number);

close(fd1);

close(fd2);

}else{

printf("INPUT SHOULD OVER 10\n");

}

}

1. **실행결과**

**3-1. draw.c**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**3-2. upperLower.c**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**3-1. randSort.c**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

1. **구현을 통해 배운 점**

draw.c, upperLower.c, randSort.c 구현을 통하여 open(), close(), read(), write(), lseek()등과 같이 C언어 프로그래밍에서 시스템 콜 함수를 이용하여 저수준 파일 입출력을 다루었다.

C언어 상의 리눅스 파일 입출력 시스템 콜을 사용하며 동작 방식을 이해하고 사용법을 익힐 수 있었다.

또 시스템 콜 함수를 이용하면 바이트 단위로 파일을 다루기 때문에 다른 고수준 파일 입출력 라이브러리를 사용하는것보다 더 빠른 속도로 동작할 수 있음을 배울 수 있었다.