REPORT

운영체제 과제 0 보고서

제출일	2022. 03. 23	전 공	산업정보시스템공학과
과 목	운영체제(OS)	학 번	201715563
담당교수	박현찬	이 름	최인혁

1. Code

```
#include <stdio.h>
#include <stdlib.h>
typedef  struct  {
    int  pid; //ID
    int  arrival_time; //도착시간
    int  code_bytes; //코드길이(바이트)
} process; //프로세스 구조체
typedef  struct {
    unsigned  char  operation; //operation
    unsigned  char  length;   //length
} code; //코드 구조체
int  main(int  argc, char * argv []){
    process p_data; //프로세스 구조체 변수 선언
    code c_data; // 코드 구조체 변수 선언

    while (fread(&p_data,sizeof (process),1 ,stdin) == 1 ){ //프로세스 구조체 크기만큼 읽어들임
        fprintf(stdout,"%d %d %d \n",p_data.pid,p_data.arrival_time,p_data.code_bytes); //프로세스 정보 출력
        for (int  j=0 ;j<p_data.code_bytes/2 ;j++){ // operation 과 lenghth가 코드 구조체안에 있기 때문에 프로세스 코드길이의 절반만큼 반복
            fread(&c_data,sizeof (code),1 ,stdin); //코드 구조체 크기만큼 읽어들임
            fprintf(stdout,"%d %d \n",c_data.operation,c_data.length); // 코드 정보 출력
        }
    }
}
```

2. Description

```
typedef  struct  {
    int  pid; //ID
    int  arrival_time; //도착시간
    int  code_bytes; //코드길이(바이트)
} process; //프로세스 구조체
typedef  struct {
    unsigned  char  operation; //operation
    unsigned  char  length;   //length
} code; //코드 구조체
```

바이너리 파일을 읽어들이기 위해 필요한 프로세스 구조체와 코드 구조체를 선언한다. 프로세스 구조체안에는 ID를 나타내는 pid변수, 도착시간을 나타내는 arrival_time변수, 코드길이를 나타내는 code_bytes 변수가 선언되어있다. 그리고 코드 구조체 내에는 operation 변수와 length 변수가 선언되어있다.

```
process p_data; //프로세스 구조체 변수 선언
    code c_data; // 코드 구조체 변수 선언

    while (fread(&p_data,sizeof (process),1 ,stdin) == 1 ){ //프로세스 구조체 크기만큼 읽어들임
        fprintf(stdout,"%d %d %d \n",p_data.pid,p_data.arrival_time,p_data.code_bytes); //프로세스 정보 출력
```

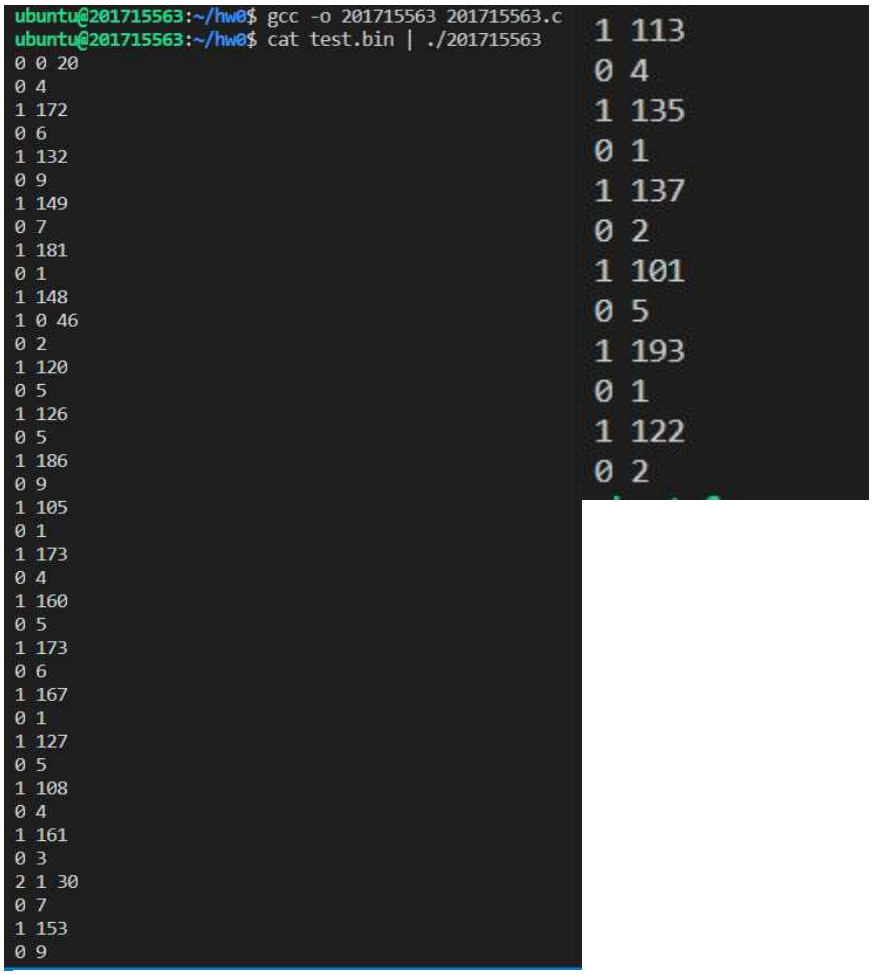
프로세스 구조체와 코드 구조체의 변수들을 선언하고 while문을 이용하여 프로세스 구조체 크기만큼 바이너리 파일에서 읽어들여 구조체 변수들을 이용해서 출력한다.

```
for (int  j=0 ;j<p_data.code_bytes/2 ;j++){ // operation 과 lenghth가 코드 구조체안에 있기 때문에 프로세스 코드길이의 절반만큼 반복
            fread(&c_data,sizeof (code),1 ,stdin); //코드 구조체 크기만큼 읽어들임
            fprintf(stdout,"%d %d \n",c_data.operation,c_data.length); // 코드 정보 출력
        }
```

프로세스 다음에 있는 코드들을 읽기위해 전 줄에서 읽었던 프로세스 코드길이의 반만큼 for문을 이용해 반복하여 코드 구조체 크기만큼 읽어들여 출력한다.

그리고 while문을 돌면서 바이너리 파일을 끝까지 읽어들인다.

3. Result



Compilation Warnings

```
oshw0c.c: In function 'main':
oshw0c.c:22:13: warning: ignoring return value of 'fread', declared with attribute warn_unused_result [-Wunused-result]
   22 |             fread(&c_data,sizeof(code),1,stdin);
      |             ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Execution Results

✔ ✔ ✔ ✔ ✔

> **Test case #1:** AC [0.005s,524.00 KB] (1/1)
> **Test case #2:** AC [0.005s,524.00 KB] (1/1)
> **Test case #3:** AC [0.005s,524.00 KB] (1/1)
> **Test case #4:** AC [0.005s,524.00 KB] (1/1)
> **Test case #5:** AC [0.005s,524.00 KB] (1/1)

Resources: 0.024s, 524.00 KB
Final score: 5/5 (10.0/10 points)

4. 고찰

이번 과제를 하면서 어려웠던 점은 딱히 없었지만 fread라는 함수에 대해서 오해를 했었다.

```
  while (fread(&p_data,sizeof (process),1 ,stdin) == 1 ){ //프로세스 구조체 크기만큼 읽어들임
        fprintf(stdout,"%d %d %d \n",p_data.pid,p_data.arrival_time,p_data.code_bytes); //프로세스 정보 출력
        for (int  j=0 ;j<p_data.code_bytes/2 ;j++){ // operation 과 lenghth가 코드 구조체안에 있기 때문에 프로세스 코드길이의 절반만큼 반복
            fread(&c_data,sizeof (code),1 ,stdin); //코드 구조체 크기만큼 읽어들임
            fprintf(stdout,"%d %d \n",c_data.operation,c_data.length); // 코드 정보 출력
        }
    }
```

위의 첫번째 while문 조건안에 있는 fread라는 함수가 프로세스 구조체만큼 바이너리 파일을 읽고 출력을 하고 두번째 for문 안에있는 fread 함수가 실행되면 다시 처음 위치부터 파일이 읽히는 줄 알고 맨 처음에 fseek 라는 함수를 써서 위치를 조정했었다.
하지만 교수님께 자문을 구하고 fseek를 쓸필요가 없다고 말씀을 하셨고 더 찾아본 결과 파일 포인터의 위치는 읽기 작업 후에 자동으로 업데이트된다는 것을 알 수 있었다.

참고 사이트:https://stackoverflow.com/questions/10696845/does-fread-move-the-file-pointer