**1. 다음 기능을 하는 프로그램(add1)을 만드시오**

**1) 데이타 화일의 마지막 숫자를 읽어서 1을 더해서 그 화일의 끝에 추가해서 기록(데이타 화일은 초기에 0만 써있다.)  
2)위 함수를 1000번 호출  
3)이 프로그램을 실행하면 0 부터 1000까지 1001개의 숫자가 기록될 것이다**

(다운로드)

#include <stdio.h>

#include <stdlib.h>

void main(){

int i;

for(i=0; i<1000; i++){

FILE \*fp = fopen("data.txt", "a+");

int data;

if( fp==NULL ){

perror("file opening err");

exit(1);

}

//while문이 끝나면 fp는 파일의 마지막을 가르키고, data에는 파일의 마지막 숫자가 저장됨

while(!feof(fp))

fscanf(fp, "%d", &data);

fprintf(fp, "%d\n", data+1);

fclose(fp);

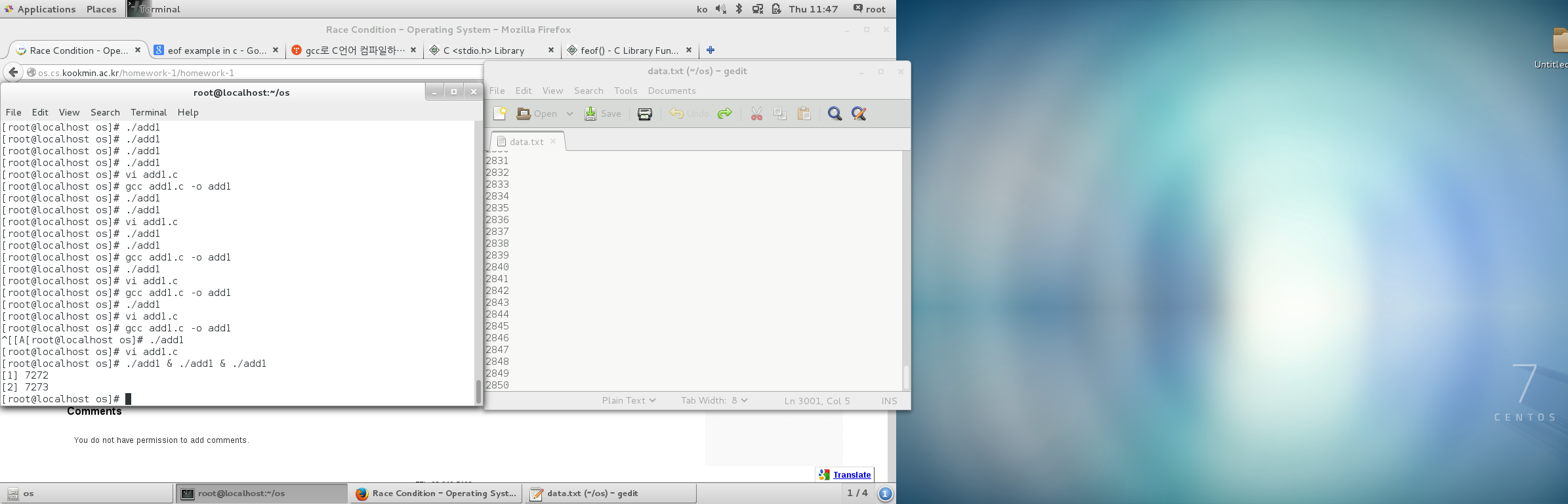
}

}

**2. 이 프로그램을 두개 이상 동시에 실행하여 결과를 살펴보시오**

**1) n개를 동시에 실행했을 때 화일의 마지막 숫자가 1000\*n인가?**

3개를 실행했는데 마지막에 3000이 안나왔음(아래 그림 참조)

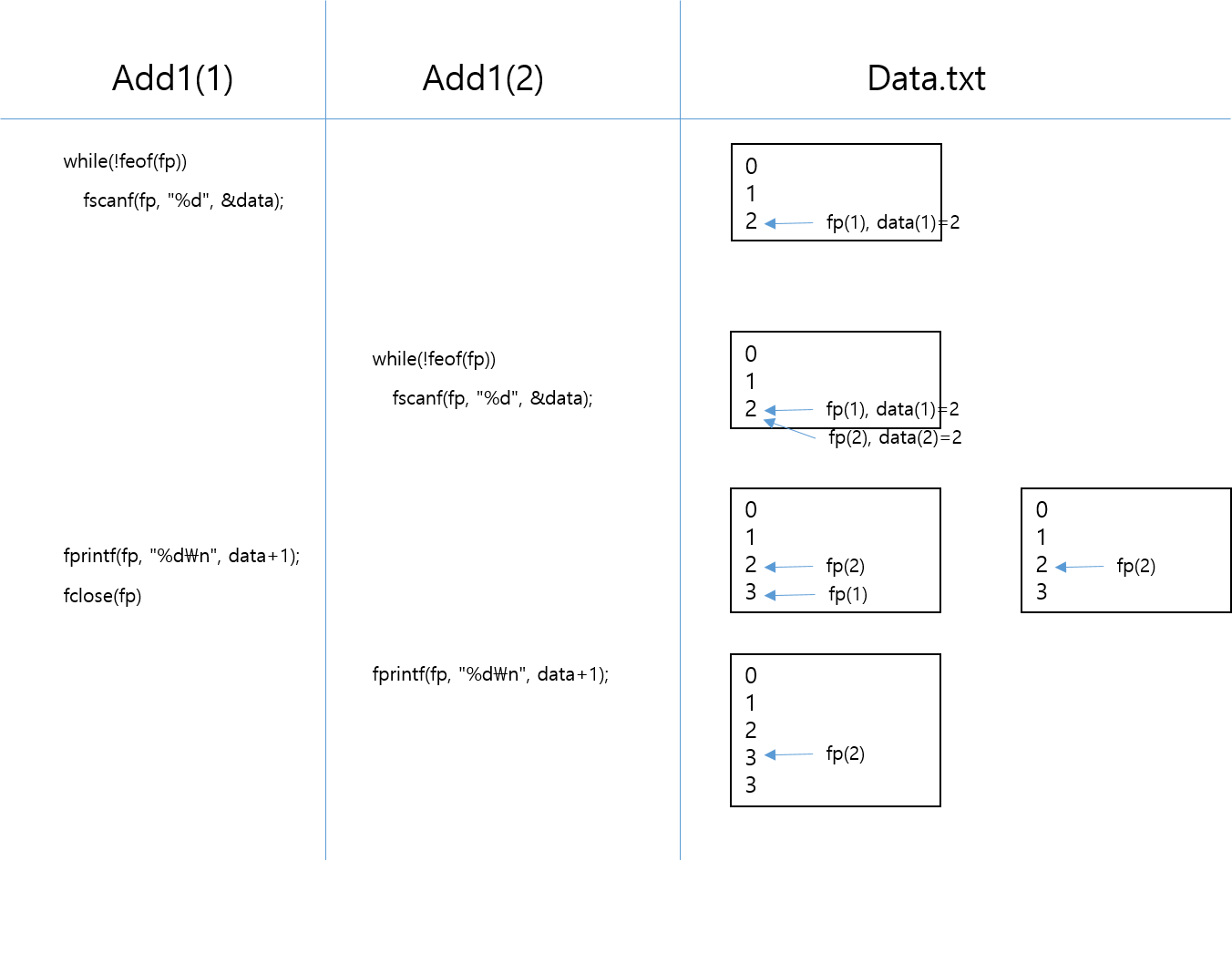


**2) 결과가 1000\*n이 아니라면 그 원인은 무엇인지 설명하시오.**

Race condition 때문에 이와 같은 일이 발생한다.

Race condition은 둘 이상의 입력/조작이 동시에 일어나는 상황을 뜻한다. Race condition에서는 event가 프로그래머 의도하지 않은 순서로 발생하지 않으면 bug를 일으킬 수 있다.

하나만 예를 들자면, 동시에 실행되는 add1(1)과 add1(2)가 있다고 할 때 다음과 같은 전개가 가능하다



이런 문제가 발생하는 이유는 두 프로세스가 실행 중 동시에 같은 데이터파일(data.txt)에 접근해 읽고 쓰기 때문이다. 따라서 한번에 하나의 프로세스만이 data.txt파일에 접근 가능 하도록 add1에서 data.txt에 접근/수정하는 code를 critical section으로 정해주어야 한다.